НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет
Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:
Завідувач кафедри
_____ Олександр КОВАЛЬ
«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Програмне забезпечення веб-технологій та мобільних пристроїв»

спеціальності 121 «Інженерія програмного забезпечення»

на тему: «Інструментальні засоби розпізнавання поведінки людини з використанням сенсорів мобільного телефону»

Виконала:
студентка IV курсу, групи ТІ-62
Мовлян Тетяна Вікторівна _____

Керівник:
Доцент, к.е.н.,
Гусєва Ірина Ігорівна _____

Консультант:
_____

Рецензент:
Доцент, к.т.н.,
Веремійчук Юрій Андрійович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ — 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки: 121 Інженерія програмного забезпечення

Спеціалізація: Програмне забезпечення веб-технологій та мобільних пристроїв

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр КОВАЛЬ
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту

Мовлян Тетяні Вікторівні
(прізвище, ім’я, по батькові)

1. Тема роботи _Інструментальні засоби розпізнавання поведінки людини з використанням сенсорів мобільного телефону

керівник роботи_____ Гусєва Ірина Ігорівна, к.е.н.
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”25” травня 2020р. № **1168-с**

2. Строк подання студентом роботи _____12.06.2020р.

3. Вихідні дані до роботи _мова програмування Dart, фреймворк Flutter, середовище розробки Visual Studio Code

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) розробити мобільний застосунок для виявлення та ідентифікації нетипової поведінки при керуванні транспортним засобом; проаналізувати існуючі методи класифікації та фільтрації; розробити метод для розпізнавання поведінки.

5. Перелік ілюстративного матеріалу

Актуальність, Постановка задачі, Стилі керування, Ідентифікація поведінки, Архітектура системи, Алгоритм виявлення та ідентифікації маневру, Діаграма класів, Діаграма розгортання, Модель бази даних, Діаграма прецедентів, Сторінки реєстрації та авторизації, Інтерфейс головної сторінки, Запис та аналіз поїздки, Сторінка статистики, Висновки.

_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання "_11_"_____жовтня_____ 2019 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	15.10.2019	
2.	Вивчення та аналіз задачі	16.10.2019-16.12.2019	
3.	Розробка архітектури та загальної структури системи	09.03.2020-23.03.2020	
4.	Розробка структур окремих підсистем	24.03.2020-10.04.2020	
5.	Програмна реалізація системи	14.04.2020-10.05.2020	
6.	Оформлення пояснювальної записки	18.05.2020-02.06.2020	
7.	Захист програмного продукту	14.05.2020	
8.	Передзахист	10.06.2020	
9.	Захист	15.06.2020	

Студент _____ _____Мовлян Т.В._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ _____Гусєва І.І._____
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

У роботі розглянуті теоретичні та практичні аспекти розробки програмного продукту для розпізнавання поведінки людини з використанням сенсорів мобільного телефону.

Метою роботи є створення мобільного застосунку для виявлення та ідентифікації нетипової поведінки при керуванні транспортним засобом.

Для досягнення мети розроблено класифікаційну модель, реалізовано алгоритми фільтрації та класифікації даних, розроблено метод розпізнавання поведінки водія та протестовано застосунок на реальних даних.

Пояснювальна записка складається зі вступу, п’яти розділів, висновку, списку використаних джерел; містить 64 сторінки, 33 рисунки, 3 таблиці та 3 додатки. Список використаних джерел включає 17 бібліографічних найменувань.

Ключові слова: мобільний застосунок, нетипова поведінка при керуванні транспортним засобом, небезпечні маневри, KNN алгоритм, SMA алгоритм.

ABSTRACT

The paper considers theoretical and practical aspects of software development for recognizing human behavior using mobile phone sensors.

The purpose of this work is to create a mobile application for detecting and identifying abnormal driving behavior.

To achieve the goal, a classification model was developed, filtering and data classification algorithms were implemented, a method for recognizing driver behavior was developed, and an application was tested on real data.

The explanatory note consists of an introduction, five chapters, conclusion, list of used sources; contains 64 pages, 33 figures, 3 tables and 3 applications. The list of used sources includes 17 bibliographic items.

Key words: mobile application, abnormal driving behavior, dangerous maneuvers, KNN algorithm, SMA algorithm.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ТЗ	—	Транспортний засіб
SMA	—	Simple Moving Average (Просте ковзне середнє)
KNN	—	k-nearest Neighbor (k-найближчих сусідів)
JS	—	JavaScript
UI	—	User interface (Користувацький інтерфейс)
HTTPS	—	HyperText Transfer Protocol Secure
SSL	—	Secure Sockets Layer (Рівень захищених сокетів)
TLS	—	Transport Layer Security (Захист на транспортному рівні)
SI	—	Safety index (Індекс безпеки)

ВСТУП

Дорожньо-транспортні пригоди є однією з серйозних загроз здоров'ю та життю людей. Згідно даних Всесвітньої організації охорони здоров'я щорічно в світі в результаті ДТП гине понад 1 млн людей та більше 50 млн зазнають травм. В Україні в 2019 році сталося 160 тис. дорожньо-транспортних пригод, в результаті яких загинуло більше 4 тис. осіб. Основними причинами ДТП є перевищення швидкості руху, порушення правил маневрування та переїзду перехресть.

Дослідження показують, що поведінка за кермом є провідною причиною виникнення понад 90% дорожньо-транспортних пригод. Моніторинг небезпечної поведінки за кермом допомагає звернути увагу водія на свій стиль керування транспортним засобом та пов'язані з цим ризики, тим самим зменшуючи відсоток необережного керування.

Майже кожен автомобіль має засоби пасивного (ремені безпеки, подушки безпеки і т. д.) та активного (система контролю швидкості, система контролю рядності руху) захисту [1]. Пасивні системи спрацьовують лише після настання ДТП. Активні системи діють завчасно, намагаючись запобігти зіткненню, але їх вартість достатньо висока.

Оскільки, згідно статистики, в світі 3.5 млрд користувачів смартфонів, все більше і більше даних можна збирати і аналізувати, щоб допомогти нам краще зрозуміти повсякденну діяльність. Поєднуючи нові методи передачі даних і покращені моделі аналізу, ми очікуємо знайти нові способи усунення основних факторів ризику і підвищення безпеки дорожнього руху.

Розроблений мобільний застосунок може використовуватися страховими компаніями. Ці компанії хочуть знати, з ким вони заключають договір. Тому з'явився термін "Розумне страхування". Страхові компанії вивчають поведінку майбутнього клієнта за даними із застосунку: як часто він порушує правила дорожнього руху, наскільки агресивно керує транспортним засобом і т. д. Залежно від цього водію призначається вартість полісу.

Ще одним можливим застосуванням даного мобільного застосунку є створення сервісу для заохочення водіїв їздити акуратно і не порушувати правила дорожнього руху. Застосунок оцінює манеру і стиль керування транспортним засобом, система обробляє ці дані і за відповідну поведінку на дорозі водію пропонують нагороду. Наприклад, нагородою може бути до 20% кешбеку від суми страхування. Даний сервіс може знизити кількість ДТП на дорогах.

Також розроблений мобільний застосунок може використовуватися сервісами для спільних поїздок. Наприклад, водій встановлює застосунок для розпізнавання поведінки. Потім система сервісу отримує дані із застосунку і формує рейтинг водія. Людина, яка має кудись поїхати, відкриває сайт чи мобільний застосунок сервісу, продивляється профайли водіїв і обирає з ким подорожувати.

Таким чином, розроблений мобільний застосунок для розпізнавання поведінки людини є актуальним та може використовуватися в багатьох галуззях.

1. ЗАДАЧА ВИЯВЛЕННЯ ТА ІДЕНТИФІКАЦІЇ НЕТИПОВОЇ ПОВЕДІНКИ ПРИ КЕРУВАННІ ТРАНСПОРТНИМИ ЗАСОБАМИ

Метою даної роботи є створення мобільного застосунку для виявлення та ідентифікації нетипової поведінки при керуванні транспортним засобом. Призначення застосунку — збереження даних поїздки, розпізнавання та ідентифікація нетипової поведінки.

Задачі, які потрібно розв’язати для досягнення мети:

1. Провести аналіз існуючих мобільних застосунків, що відстежують поведінку водія для попередження ДТП або пом’якшення їх наслідків.
2. Провести аналіз збережених “еталонних” поїздок та виокремити характерні ознаки для класифікації небезпечних маневрів.
3. Розробити класифікаційну модель для ідентифікації нетипової поведінки за кермом.
4. Проаналізувати існуючі методи класифікації.
5. Проаналізувати існуючі методи фільтрації даних.
6. Розробити метод розпізнавання поведінки водія при керуванні транспортним засобом.
7. Протестувати розроблену систему на реальних даних.

Вимоги до розроблюваних програмних засобів:

- Можливість створення облікового запису та зміни особистої інформації.
- Система виявлення та ідентифікації нетипової поведінки має складатися з наступних підсистем: підсистема виявлення початку і кінця нетипової поведінки; підсистема класифікації нетипової поведінки; підсистема розпізнавання поведінки водія.
- Вхідними даними підсистеми виявлення початку і кінця нетипової поведінки мають бути дані поїздки (показання сенсорів).

- Вихідними даними підсистеми виявлення початку і кінця нетипової поведінки мають бути індекси початку і кінця кожної нетипової поведінки.
- Вхідними даними підсистеми класифікації нетипової поведінки мають бути індекси початку і кінця нетипової поведінки.
- Вихідними даними підсистеми класифікації нетипової поведінки має бути вид нетипової поведінки.
- Вхідними даними підсистеми розпізнавання поведінки водія має бути кількість небезпечних маневрів за кожну хвилину поїздки.
- Вихідними даними підсистеми класифікації нетипової поведінки має бути модель поведінки водія.
- Можливість зберігати дані поїздки.
- Можливість переглядати статистику всіх поїздок і окремої поїздки.

2. ОПИС ЕТАПІВ РОЗПІЗНАВАННЯ НЕТИПОВОЇ ПОВЕДІНКИ

2.1. Аналіз існуючих програмних рішень розпізнавання поведінки

CarSafe — мобільний застосунок допомоги водію, який використовує алгоритми комп'ютерного зору і машинного навчання для відстеження втоми і ослабленої уваги водія за допомогою фронтальної камери смартфона та для стеження за дорожньою обстановкою за допомогою основної камери.

Передня камера відстежує положення голови водія, а також кількість кліпань очима за одну хвилину для виявлення періодів мікросну, втоми чи сонливості. Коли виявляється один з цих станів, CarSafe попереджає водія, зображуючи на екрані іконку з чашкою кави разом зі звуковим сповіщенням.

Задня камера відстежує відстань між автомобілями, щоб визначити, чи їде водій занадто близько до машини спереду, а також для відстеження зміни смуги руху. Якщо застосунок виявить, що водій рухається занадто близько до машини спереду, кольоровий рядок на сенсорному екрані змінює колір із зеленого на червоний разом зі звуковим попередженням.

Перевагами цього застосунку є використання багатьох сенсорів, вбудованих в мобільний телефон (основна та фронтальна камери, GPS, акселерометр, гіроскоп), що надають високу точність визначення стану втоми, відволікання, наближення до передньої машини. Однак на даний момент застосунок CarSafe не доступний для тестування, що є недоліком.

Augmented Driving — додаток доповненої реальності, що використовує виключно основну камеру смартфона для відстеження дорожньої обстановки та попередження водія про можливе настання ДТП за допомогою звукових сповіщень.

Цей додаток пропонує такі функції для водія: повідомлення про недотримання дистанції до ТЗ, що рухається попереду; стеження за дорожньою розміткою та повідомлення про перевищення швидкості ТЗ.

Перевагами цього застосунку є відстеження основних небезпечних типів поведінки за кермом, що можуть призвести до настання ДТП, відображення відео поточної обстановки на дорозі з елементами доповненої реальності, що показують швидкість транспортного засобу, напрямок руху та відстань до найближчих машин. Головні недоліки Augmented Driving — застосунок доступний для використання лише на iOS-пристроях; більшість функції є платними; через те, що увесь час відображається відео, швидко витрачається заряд акумулятора.

Застосунок Nexar — AI Dashcam доступний на платформах iOS та Android, працює як відеореєстратор, який зберігає інформацію про поїздку: час, координати, швидкість автомобіля і запис ДТП. Nexar веде запис відеопотоку одночасно з фронтальної і основної камер смартфона, зберігає номерні знаки ТЗ, що порушують правила дорожнього руху, щоб згодом при появі порушника в зоні видимості камери попередити водія за допомогою текстових і звукових сповіщень. Також застосунок повідомляє, якщо десь на маршруті, яким прямує транспортний засіб сталася дорожньо-транспортна пригода, щоб користувач був обачнішим і за можливості змінив маршрут.

Перевагами застосунку Nexar — AI Dashcam є те, що він попереджає користувача, якщо десь поруч порушник, або десь на маршруті слідування ДТП і необхідно бути уважнішим. Недоліком є те, що зберігається відео всієї поїздки, а не окремих небезпечних маневрів і тому застосунку потрібно багато пам’яті.

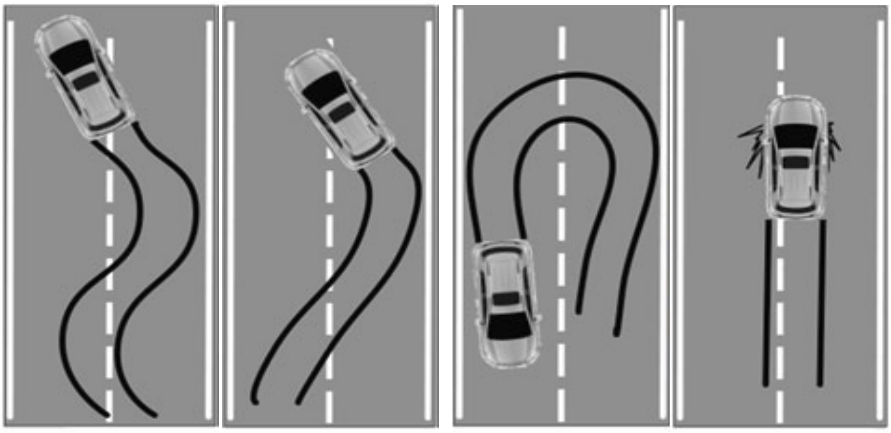
Штрафи UA — український застосунок, що спостерігає, наскільки безпечно керують автівкою: якої швидкості дотримуються, чи використовують телефон за кермом. Після цього додаток виставляє водієві оцінку і пропонує відповідну знижку на страхування. Також, в цій системі можна сплачувати штрафи за паркування, порушення правил дорожнього руху.

Перевагою цього застосунку є функція “Безпечне керування”, що моніторить поведінку водія при керуванні транспортним засобом. Ця функція є мотивацією бути

більш обачним і уважним за кермом, адже за безпечне керування можна отримати бонуси (кешбек на страхування чи знижку на пальне). Недоліком даного застосунку є те, що на даний час він доступний тільки на iOS.

2.2. Аналіз моделей нетипової поведінки при керуванні транспортним засобом

В результаті аналізу визначено чотири види нетипової поведінки водія, які наведені на рисунку 2.1.



а) б) в) г)

Рисунок 2.1 — Види нетипової поведінки водія: а — S-подібний рух, б — Різке перестроювання, в — Швидкий розворот, г — Різке гальмування

S-подібний рух (рисунок 2.1а) — рух, при якому транспортний засіб переміщується по черзі з однієї смуги руху в іншу, тобто S-траєкторією; Різке перестроювання (рисунок 2.1б) — рух, при якому відбувається різка зміна смуги руху; Швидкий розворот (рисунок 2.1в) представляє собою швидкий поворот в U-подібній формі, тобто швидкий поворот на 180° і рух в протилежному напрямку; Різке гальмування (рисунок 2.1г) відбувається при різкому натисканні на гальма, при цьому за дуже короткий проміжок часу швидкість транспортного засобу різко падає [3].

Для визначення положення пристрою в просторі використовуються акселерометр та гіроскоп — сенсори, що вбудовані в усі сучасні мобільні телефони.

Основним призначенням акселерометра є надання інформації про поточне прискорення пристрою, тобто різниці прискорення пристрою і прискорення вільного падіння. У стані спокою показання сенсора збігаються з вектором прискорення вільного падіння [4].

Гіроскоп вимірює кутову швидкість. Зазвичай використовується спільно з акселерометром для відстеження змін в рухах. У мобільних пристроях використовуються сенсори кутової швидкості. Цей тип гіроскопів є набагато простішим і дешевшим при досить високій точності.

Комбінація акселерометра і гіроскопа дозволяє відстежувати і фіксувати рух в тривимірному просторі.

Для виявлення руху автомобіля смартфон встановлюється в центрі приладової панелі автомобіля, осі смартфона (X_p, Y_p, Z_p) збігаються з осями автомобіля (X_c, Y_c, Z_c), як показано на рисунку 2.2. Трьохосьовий акселерометр вимірює величину прискорення сили в $м/с^2$, яка застосовується до транспортного засобу за всіма трьома фізичними осями (X, Y, Z). Показання трьох осей акселерометра позначаються як acc_x, acc_y, acc_z [5].

Рисунок 2.2 — Система координат смартфона та автомобіля

Гіроскоп вимірює швидкість обертання транспортного засобу в $рад/с$ навколо трьох фізичних осей (X, Y, Z). Показання гіроскопа позначаються як ori_x, ori_y, ori_z, ці свідчення є показником повороту транспортного засобу [6].

Нехай acc_x — це прискорення по осі X, acc_y — прискорення по осі Y (дані акселерометра), ori_x — це орієнтація по осі X, ori_y — це орієнтація по осі Y (дані гіроскопа).

S-подібний рух (Weaving). На рисунку 2.3а показані схеми прискорення і орієнтації при S-подібному русі за даними акселерометра і гіроскопа. На рисунку ми спостерігаємо різке коливання по acc_x, і це коливання триває деякий період часу, в той час як acc_y залишається гладкою. Таким чином, як стандартне відхилення, так і діапазон acc_x дуже великий, а тривалість значна. Середнє значення acc_x приблизно дорівнює нулю. Крім того, значення орієнтації мають такий самий характер, як і значення прискорення.

Різке перестроювання (Swerving). На рисунку 2.3б показані графіки зміни прискорення і орієнтації різкого перестроювання. Дана поведінка є раптовою, тому є дуже короткою за тривалістю. Коли відбувається різке перестроювання спостерігається великий пік як на acc_x, так і ori_x. Діапазон і стандартне відхилення як acc_x, так і ori_x великі. Крім того, як acc_y, так і ori_y майже не змінюються.

Швидкий розворот (Fast U-turn). На рисунку 2.3в показані схеми прискорення і орієнтації швидкого розвороту. Коли водій швидко повертає вправо або вліво в U-подібній формі, acc_x швидко зростає до високого значення або швидко падає до дуже низького значення, відповідно. Стандартне відхилення acc_x є великим на початку і в кінці швидкого розвороту, середнє значення acc_x далеке від 0 і діапазон acc_x великий. Щодо acc_y, тут очевидних змін немає. Більш того, ori_x буде проходити через нульову точку. А саме, ori_x буде змінюватися від позитивного до негативного або від негативного до позитивного, в залежності від вихідного напрямку руху. Отже, стандартне відхилення і діапазон значень ori_x будуть великими. Середні значення ori_x в першій половині і другій половині будуть протилежного знаку. Для завершення швидкого розвороту потрібен деякий час, тому його тривалість є значною.

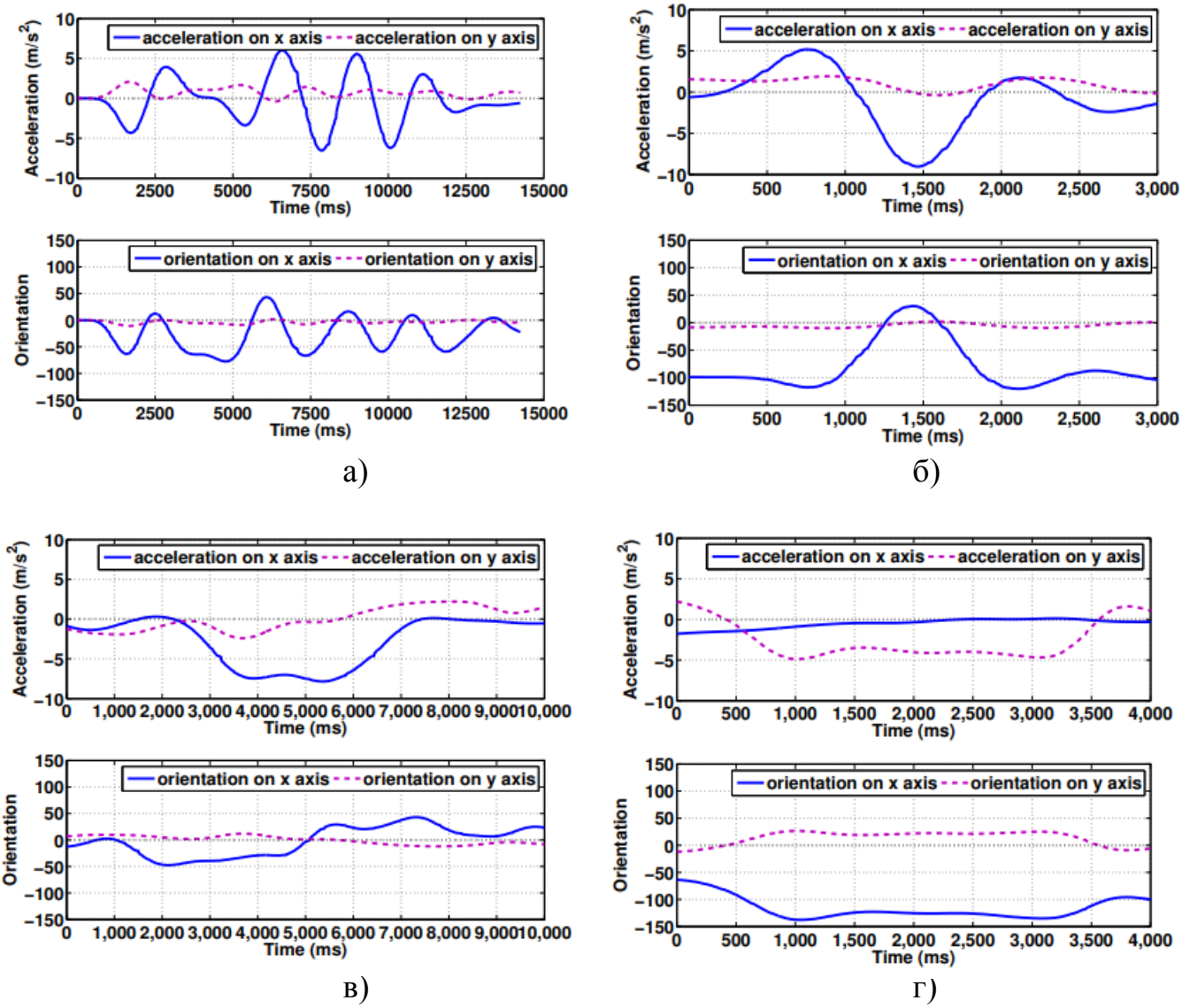


Рисунок 2.3 — Моделі прискорення та орієнтації чотирьох видів нетипової поведінки при керуванні: а — S-подібний рух, б — Різке перестроювання, в — Швидкий розворот, г — Різке гальмування

Різке гальмування (Sudden braking). На рисунку 2.3г зображені моделі прискорення і орієнтації при різкому гальмуванні. Коли транспортний засіб раптово гальмує, acc_x залишається на попередньому рівні, в той час як acc_y різко падає і деякий час залишається негативним. Таким чином, стандартне відхилення і діапазон значень acc_x невеликий. Що стосується acc_y, стандартне відхилення є великим на початок і кінець різкого гальмування . Крім того, немає ніяких очевидних змін як на ori_x, так і на ori_y. Оскільки раптове гальмування є різким, тривалість його — коротка.

Типова (неагресивна) поведінка при керуванні транспортним засобом. Типова поведінка при керуванні ТЗ означає гладкий і безпечний рух з невеликими коливаннями. Оскільки радикальних дій в нормальній поведінці при керуванні небагато, то значення як acc_x так і acc_y не дуже великі. Так середнє, стандартне відхилення, максимальне і мінімальне значення прискорення по осі X і Y близькі до нуля. [3].

Зважаючи на вищенаведений аналіз, виявлено, що кожний вид керування транспортним засобом має свої унікальні характеристики, такі як стандартне відхилення, максимальне, мінімальне та середнє значення, діапазон значень на acc_x, acc_y, ori_x і ori_y, а також тривалість маневру. Тому будемо використовувати ці характеристики для визначення певних видів нетипової поведінки.

2.3. Методи обробки даних

Як правило, необроблені дані сенсорів завжди спотворені шумом, що призводить до неточності вимірювань. Наприклад, похибка даних акселерометра іноді досягає $0.05g$, тому потрібна боротьба з шумом. Є багато підходів згладжування і фільтрації даних. Нижче розглянуті деякі з них.

1) Експоненціальний фільтр

Експоненціальне згладжування — це фільтр, який послідовно отримує члени початкового ряду та повертає значення експоненціальної середньої за формулою (2.1):

$$S_t = \alpha \times x_t + (1-\alpha) \times S_{t-1}, \tag{2.1}$$

де S_t — згладжений ряд;

x_t — вхідне значення;

α — коефіцієнт фільтра, що може приймати значення від 0 до 1.

2) Фільтр ковзного середнього

Метод простого ковзного (рухомого) середнього (*англ. Simple Moving Average, SMA*) один з найпростіших методів фільтрації шуму [7]. З математичної точки зору цей метод є середнім арифметичним, а для розрахунку його значення використовується формула (2.2):

$$SMA_t = \frac{1}{n} \times \sum_{i=0}^{n-1} P_{t-i}\ , \qquad (2.2)$$

де SMA_t — значення рухомого середнього в момент часу *t*;

n — інтервал згладжування;

P_{t-i} – початкове значення в момент *(t-i)*.

3) Експоненціальне ковзне середнє

Експоненціальне ковзне середнє (*англ. Exponential Moving Average, EMA*) є одним з різновидів простого ковзного середнього і частинним випадком зваженого ковзного середнього. EMA розраховується за формулою 2.3:

$$EMA_t = \alpha \times P_t + (1-\alpha) \times EMA_{t-1}, \qquad (2.3)$$

де EMA_t — значення експоненціального ковзного в момент часу *t*;

α — ваговий коефіцієнт, $\alpha \in [0;1]$;

P_t – початкове значення в момент часу *t*;

EMA_{t-1} — значення експоненціального ковзного в момент часу *(t-1)*.

Для розрахунку оптимального значення α не існує математичної формули. На практиці для визначення значення α використовується формула (2.4) [8]:

$$\alpha = \frac{2}{n+1}, \qquad (2.4)$$

де n — інтервал згладжування.

Для розрахунку значення експоненціального ковзного середнього в момент часу *t* необхідно знати його значення в попередній момент часу (*t-1*). При цьому, в якості першого значення береться просте ковзне середнє з таким самим інтервалом згладжування.

У роботах по розпізнаванню нетипової поведінки водіїв застосовуються наступні методи класифікації: дерева рішень, метод k-найближчих сусідів, методи глибокого навчання. Переваги та недоліки даних методів представлені в таблиці 2.1.

Таблиця 2.1. Порівняння методів класифікації

Метод	Переваги	Недоліки
Дерева рішень	Проста реалізація, інтерпретація і відсутність підготовки даних для їх подальшого використання. Робота з категоріальним і інтервальними змінними. Робота з великим обсягом інформації.	Відсутність оптимальності дерева рішень в цілому, необхідність регулювання його довжини. Надлишок даних і погана читабельність.
Метод k-найближчих сусідів (алгоритм KNN)	Простота реалізації, робота з великою кількістю даних.	Необхідність початкового визначення точного числа класів.
Методи глибокого навчання	Можливість оновлювати новими даними за допомогою пакетного поширення. Архітектура може бути адаптована до багатьох типів проблем.	Вимагають дуже великого обсягу даних, значних обчислювальних ресурсів і досвіду для налаштування

Виходячи з даної таблиці і результатів досліджень, для розпізнавання нетипової поведінки водіїв будемо використовувати алгоритм KNN, який дозволяє класифікувати в умовах надходження даних в реальному часі, забезпечує впевнену класифікацію і має високу точність (> 92 %) [9].

Метод k-найближчих сусідів (KNN) — один з найпростіших алгоритмів класифікації, який також іноді використовується в задачах регресії.

Завдання класифікації в машинному навчанні — це завдання віднесення об’єкта до одного із заздалегідь визначених класів на підставі його формалізованих ознак. Кожен з об’єктів у цьому завданні представляється у вигляді вектору в N-вимірному просторі, кожний вимір в якому є описом однієї з ознак об’єкта.

Для навчання класифікаційної моделі необхідно мати набір даних, для яких вже визначено клас. Цей набір даних називається навчальною вибіркою.

Для класифікації кожного об’єкта тестової вибірки потрібно виконати наступні кроки [10]:

1. Необхідно задати кількість найближчих сусідів (k).

При значенні $k = 1$ алгоритм втратить узагальнюючу здатність, а при великих значеннях *k* більшість локальних особливостей не будуть виявлені.

2. Знайти відстань до кожного з об’єктів навчальної вибірки.

Для впорядкованих значень атрибутів знаходять Евклідову відстань за формулою (2.5):

$$D_E = \sqrt{\sum_i^n (x_i - y_i)^2}\ , \qquad (2.5)$$

де n — кількість атрибутів.

3. Вибрати k об’єктів навчальної вибірки, відстань до яких мінімальна.

4. Класифікувати тестовий набір даних.

Коли знайдено об’єкти в навчальному наборі, найбільш схожі на тестовий набір, необхідно вирішити, як вони впливають на його клас. Для цього використовується функція поєднання. Є два основних варіанти цієї функції — просте незважене голосування і зважене голосування.

a. Просте незважене голосування

Всі мають рівні права у визначенні класу. Кожен запис голосує за клас, до якого належить. Тестовому набору присвоюється клас, що набрав найбільшу кількість голосів. Якщо декілька класів отримали однакову кількість голосів, то цю проблему вирішує зважене голосування.

b. Зважене голосування

При зваженому голосуванні враховується також і відстань до нового запису. Голос за клас знаходиться за формулою (2.6):

$$votes(class) = \sum_{i=1}^{n} \frac{1}{d^2(X, Y_i)}, \qquad (2.6)$$

де $d^2(X, Y_i)$ — квадрат відстані від відомого запису Y_i до нового X;

n — кількість відомих записів класу;

class — назва класу.

Тестовому набору присвоюється клас, що набрав найбільшу кількість голосів.

2.4. Вибір атрибутів для розпізнавання та ідентифікації нетипової поведінки

Кожна поведінка за кермом має свої унікальні показники acc_x, acc_y, ori_x, ori_y та часу. Основна відмінність між різними типами поведінки при керуванні транспортним засобом полягає в максимумі, мінімумі, діапазоні значень, середніх значеннях і стандартних відхиленнях показників сенсорів (acc_x, acc_y, ori_x, ori_y) та в тривалості окремої поведінки [11].

Для вибору ефективних ознак ми проаналізували тренувальні поїздки. Рисунок 2.4 показує різницю між нормальною і нетиповою поведінкою водія в двовимірному масиві властивостей (в діапазоні значень $range_{acc,x}$ і $range_{acc,y}$). Видно, що ці дві ознаки можуть чітко розрізняти нормальну і агресивну поведінку при керуванні ТЗ. Тому нам вдається відрізняти нетипову поведінку від типової тільки за двома параметрами.

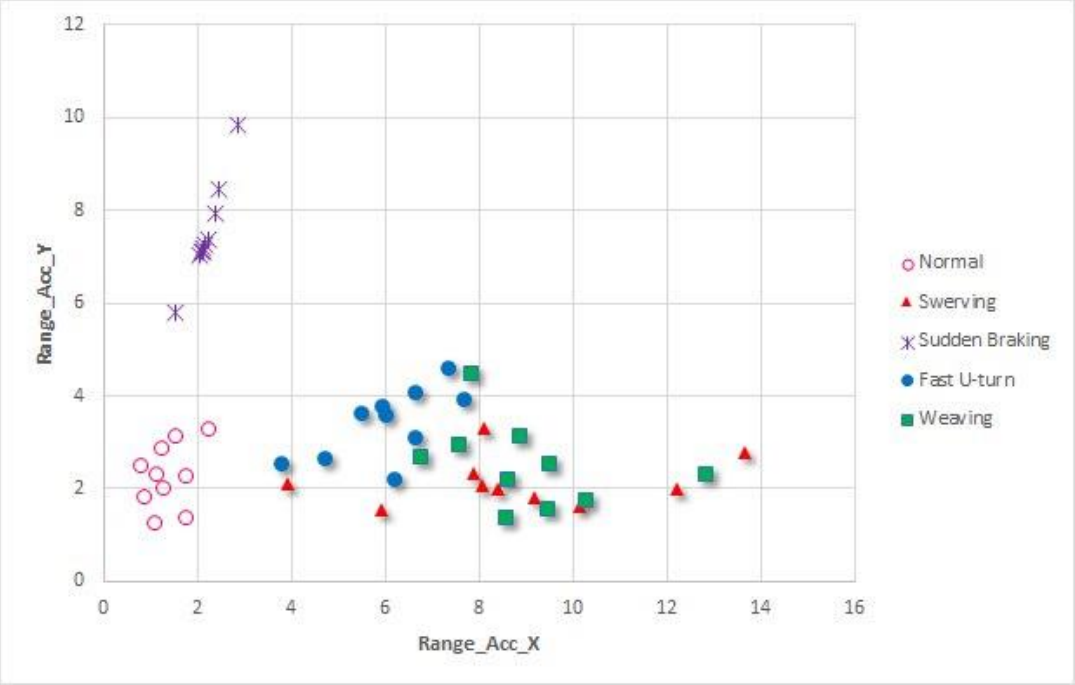


Рисунок 2.4 — Дві ознаки для розрізнення нетипової і типової поведінки

Хоч ми можемо відрізнити нетипову поведінку від типової, використовуючи двомірний масив елементів, ми не зможемо розрізнити чотири типи агресивної

поведінки один від одного тільки за допомогою двовимірних елементів. Як приклад, на рисунку 2.4, три види нетипової поведінки водія змішуються між собою.

Незважаючи на те, що всі чотири небезпечні маневри не можна відрізнити один від одного одночасно, будь-які два з них можуть бути ідентифіковані за певними двома ознаками. На рисунку 2.5 S-подібний рух (weaving) і різке перестроювання (swerving) можна розрізнити один від одного, використовуючи стандартне відхилення осі *x* акселерометра ($St\_Deviation\_Acc\_X$) та осі *y* гіроскопа ($St\_Deviation\_Ori\_Y$).

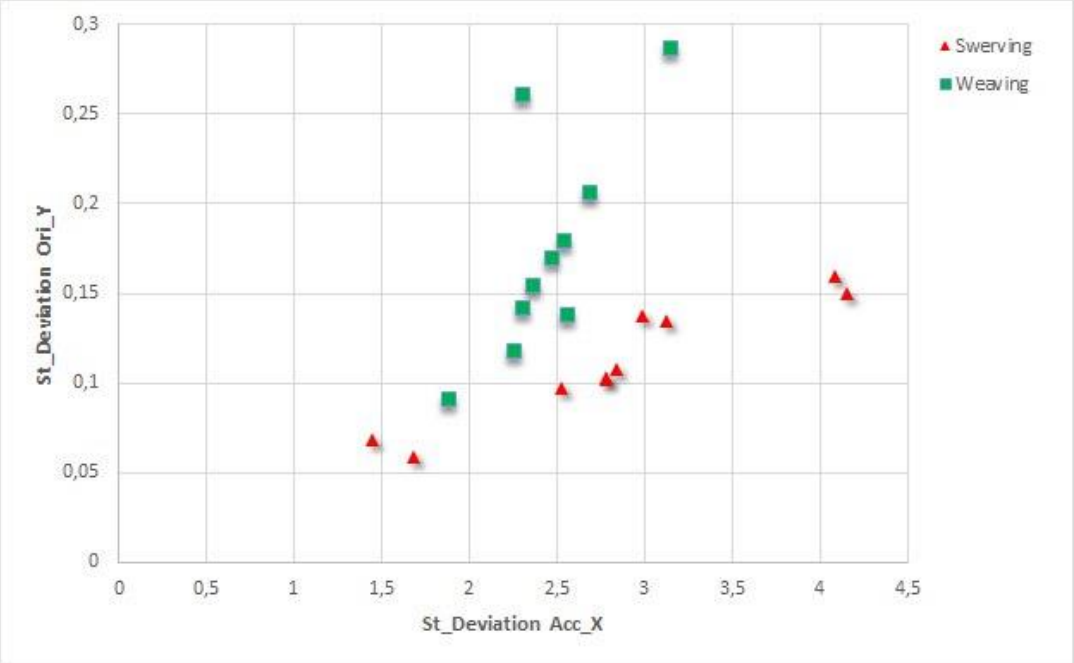


Рисунок 2.5 — Порівняння S-подібного руху та перестроювання.

Аналогічним чином, інші види нетипової поведінки також можуть бути попарно розрізнені, використовуючи дві ознаки.

На рисунку 2.6 зображено порівняння S-подібного руху (weaving) і різкого гальмування (sudden braking) з використанням пари ознак $Range\_Acc\_X$ та Min_Acc_Y.

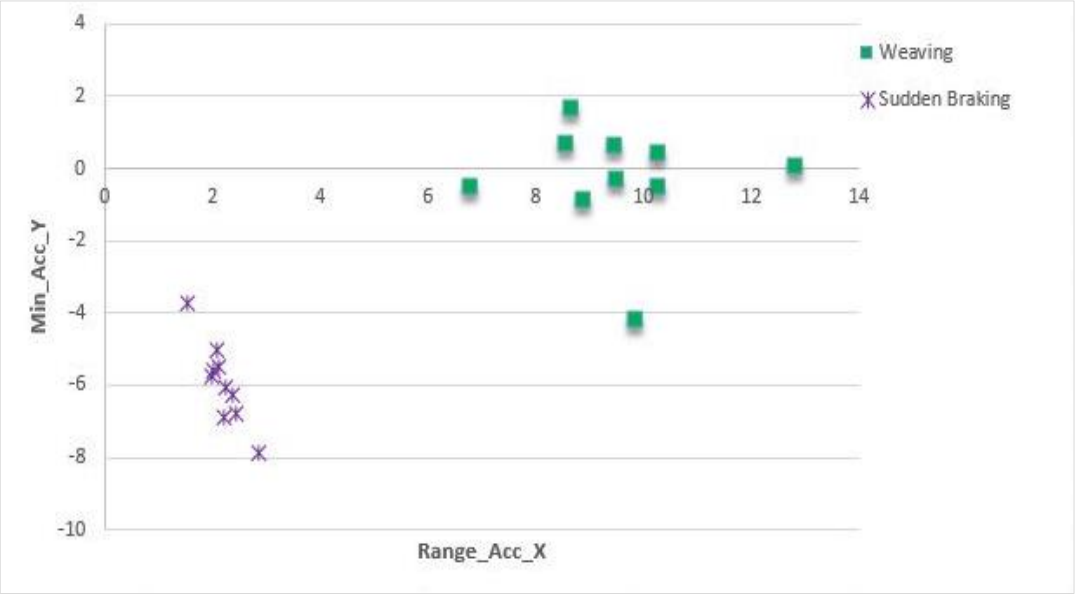


Рисунок 2.6 — Порівняння S-подібного руху та різкого гальмування

На рисунку 2.7 відображено графіки таких маневрів, як S-подібний рух та швидкий розворот (fast U-turn) з використанням пари атрибутів ($Mean\_Acc\_X$; Max_Ori_Y).

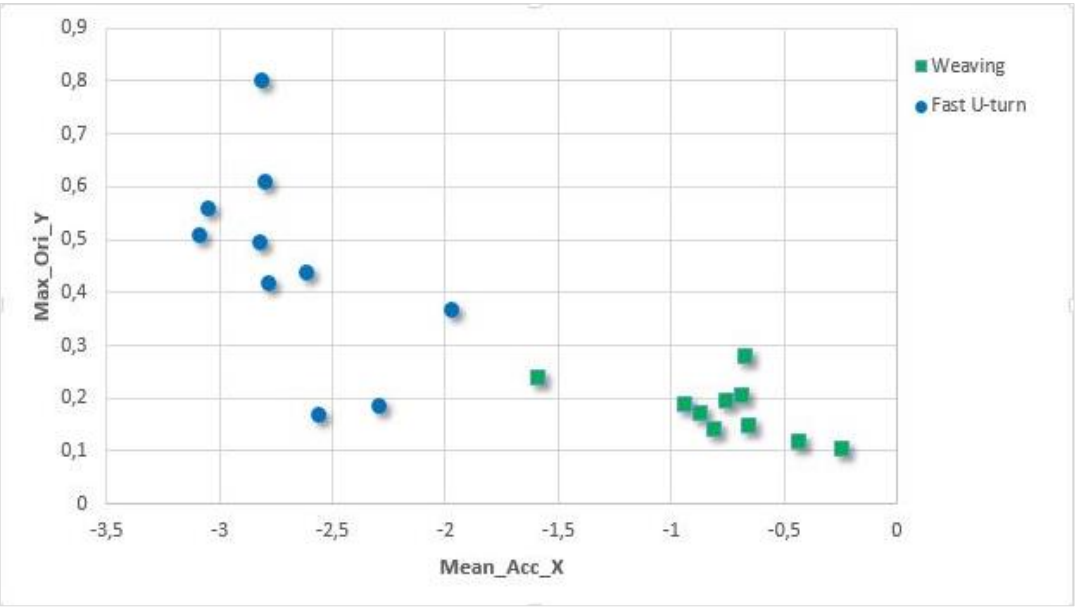


Рисунок 2.7 — Порівняння S-подібного руху та швидкого розвороту

Таким чином, на основі здійснених тестових поїздок досліджено всі можливі парні випадки. У кожному випадку, виділено кілька атрибутів, що допомагають розрізняти різні види нетипової поведінки при керуванні транспортним засобом. Всього визначено 16 ефективних характеристик, які відображають їх закономірності. Їх опис наведено в таблиці 2.2.

Таблиця 2.2 Опис характеристик для ідентифікації класу нетипової поведінки

Характеристика	Опис
$range_{acc,x}$	Різниця максимального і мінімального значення acc_x
$range_{acc,y}$	Різниця максимального і мінімального значення acc_y
$\sigma_{acc,x}$	Стандартне відхилення acc_x
$\sigma_{acc,y}$	Стандартне відхилення acc_y
$\sigma_{ori,x}$	Стандартне відхилення ori_x
$\sigma_{ori,y}$	Стандартне відхилення ori_y
$\mu_{acc,x}$	Середнє значення acc_x
$\mu_{acc,y}$	Середнє значення acc_y
$\mu_{ori,x}$	Середнє значення ori_x
$\mu_{ori,y}$	Середнє значення ori_y
$\mu_{acc,x,1}$	Середнє значення першої половини acc_x
$\mu_{acc,x,2}$	Середнє значення другої половини acc_x
$max_{ori,x}$	Максимальне значення ori_x
$max_{ori,y}$	Максимальне значення ori_y
$min_{ori,x}$	Мінімальне значення ori_x
t	Тривалість

Ці значення використовуються як атрибути для тренування моделі в офлайн частині системи (Моделювання поведінки водія).

2.5. Визначення початку і кінця маневру

Після отримання моделі класифікатора ми можемо виявити і ідентифікувати нетипову поведінку в реальному середовищі з використанням моделі. Для ідентифікації маневру тестової поїздки з використанням моделі необхідно визначити його початок і кінець. На рисунку 2.8 зображені дані акселерометра і сенсору орієнтації смартфона по осі X і осі Y за 1 хв. руху, які містять S-подібний рух (weaving).

Спосіб визначення початку і кінця нетипової поведінки пропонується на основі аналізу моделей прискорення і орієнтації для всіх типів поведінки. Коли починається нетипова поведінка, стандартне відхилення значень прискорення або орієнтації різко зростає і зберігає високе значення до кінця маневру, в той час як при нормальному керуванні стандартне відхилення завжди є малим і майже не змінюється [12].

У реальних умовах керування ми отримуємо показання від акселерометру і гіроскопу, на основі яких обчислюємо стандартне відхилення, а також середнє значення в невеликому вікні. При нормальному керуванні, система порівнює стандартне відхилення і середнє значення з деякими пороговими значеннями, щоб визначити чи починається нетипова поведінка. Розмір вікна і порогові значення можуть бути отримані із зібраних даних. Після того, як визначений початок деякого небезпечного маневру, система продовжує перевіряти стандартне відхилення і середнє значення для визначення закінчення маневру.

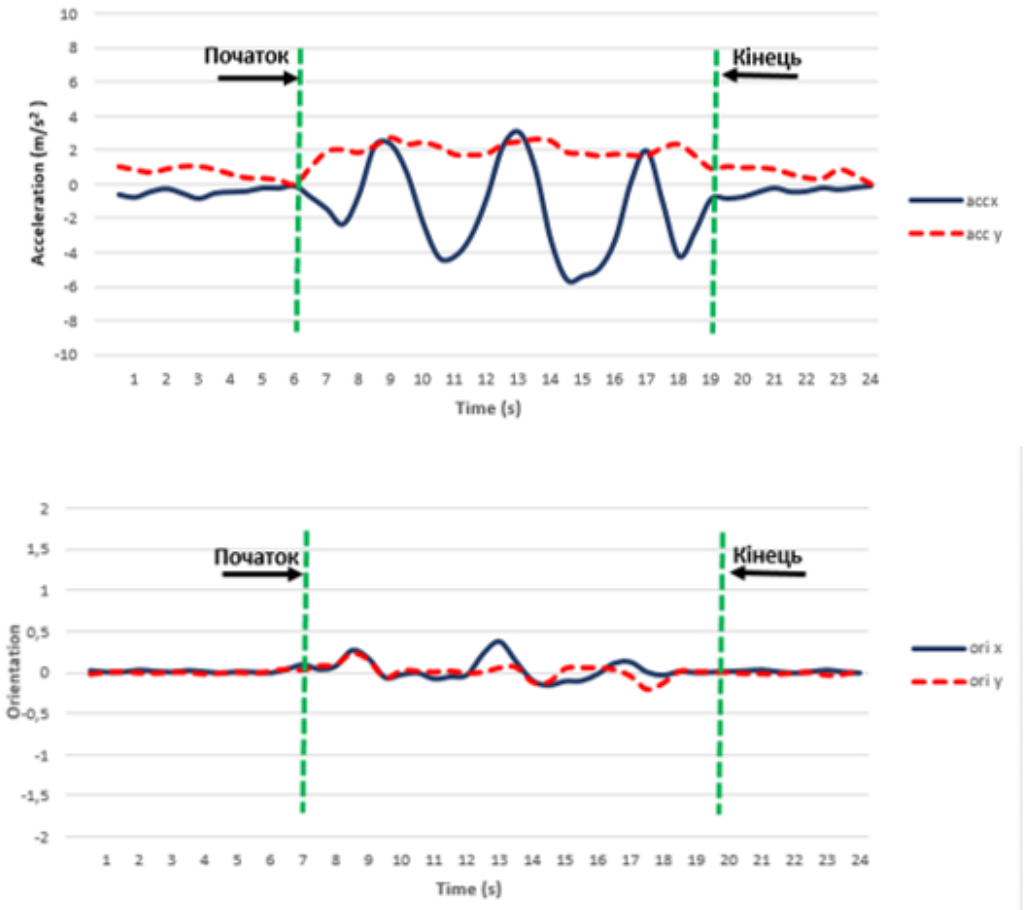


Рисунок 2.8 — Розпізнавання початку та кінця маневру

2.6. Висновки до розділу

У даному розділі наведено опис існуючих мобільних застосунків, що відстежують поведінку водія за кермом, стежать за дорожньою обстановкою і допомагають водію уникнути настання ДТП.

Описані і проаналізовані види нетипової поведінки, що розпізнаються в розробленому мобільному застосунку; описано сенсори, що використовуються; розглянуті методи фільтрації та класифікації даних. Також описано вибір атрибутів для ідентифікації нетипової поведінки; наведено алгоритм визначення початку і кінця маневру.

3. ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ РОЗПІЗНАВАННЯ ПОВЕДІНКИ

3.1. Обґрунтування вибору мови програмування

У наш час для розробки мобільних застосунків використовується один з двох популярних фреймворків — Flutter чи React Native.

Flutter — сучасний фреймворк, представлений компанією Google в 2017 році. Використовується для створення веб-застосунків, мобільних застосунків для платформ Android та iOS, графічних застосунків для операційних систем Windows, Mac та Linux. Особливістю роботи з Flutter є те, що застосунки для різних платформ можуть мати один і той самий код. Мовою розробки на Flutter є Dart. При побудові застосунку Flutter транслює код, написаний мовою Dart в нативний код застосунку, який можна запускати на будь-якій з платформ, зазначених вище [13].

Dart — об’єктно-орієнтована мова програмування від компанії Google для розробки швидких застосунків для будь-якої платформи. Всі значення, які використовуються в програмі на мові Dart є об’єктами. Входом в програму є функція main(), що обов’язково має бути присутньою в програмі [14].

React Native — це JavaScript-фреймворк для створення нативних iOS- і Android-застосунків. В його основі лежить JavaScript-бібліотека React, розроблена в Facebook. React Native орієнтований на мобільні платформи, його можна використовувати для написання чистих, швидких мобільних додатків, не залишаючи комфорту звичного фреймворка і єдиної кодової бази JavaScript.

Крім того, React Native надає «розумну» систему повідомлень про помилки і стандартні інструменти налагодження JavaScript, що полегшує процес мобільної розробки.

JavaScript (JS) — динамічна, багаторівнева мова програмування, яка відповідає специфікації ECMAScript. JS підтримує об'єктно-орієнтовані, імперативні та

декларативні стили програмування. Зараз ця мова використовується як на стороні браузера (front-end), так і на стороні сервера (back-end).

Порівняємо Flutter та React Native.

Обидва фреймворки достатньо легко встановити і розпочати роботу. У React Native більша бібліотека UI компонентів, ніж у Flutter. Проте, якщо брати до уваги лише “рідні” компоненти, то у Flutter набагато більше бібліотек. React Native залежить від багатьох сторонніх бібліотек, що в деяких випадках може виявитися проблемою, адже в них бувають помилки чи несумісності UI з конкретною платформою. Flutter має бібліотеку для швидкого доступу до сенсорів мобільного телефону, що є перевагою для розробки мобільного застосунку для розпізнавання нетипової поведінки з використанням сенсорів.

Щодо швидкості виконання, React Native використовує міст JavaScript для інтерпретації UI компонентів для рендерінгу, а потім викликає Objective-C або Java API, щоб відобразити відповідний компонент. Цей міст – додатковий рівень абстракції, що сповільнює роботу застосунку. Перевагою Flutter є те, що він використовує зовсім інший підхід до рендерінгу. Flutter компілює увесь код Dart в нативний ARM код, що обробляється безпосередньо процесором. Це дозволяє застосункам, що створені з використанням технології Flutter працювати швидше.

Безсумнівною перевагою фреймворку Flutter є велика кількість інструментів для розробки:

- Widget Inspector — навігація по дереву віджетів, що дозволяє легко керувати великою кількість вкладених компонентів.
- Hot Reload — швидка перебудова застосунку (за декілька секунд), навіть якщо всі елементи застосунку були замінені на інші.
- Debug Paint — відображення всіх відступів і маркерів позиціонування для швидкого розуміння що саме впливає на положення віджету на сторінці.
- Repaint Rainbow — райдуга навколо елементів, які перемальовуються. Це дозволяє відслідковувати небажану активність.

Також Flutter має зрозумілу документацію і багато офіційних відео-уроків з поясненнями для чого потрібен той чи інший віджет, що дозволяє швидко розібратися і знайти потрібну інформацію.

3.2. Обґрунтування вибору системи керування базою даних

У сучасному світі є дві моделі баз даних — реляційна (SQL) та нереляційна (NoSQL). Вони відрізняються архітектурою, типами даних та способом збереження інформації. Нереляційний спосіб структуризації даних полягає в позбавленні обмежень при зберіганні і використанні інформації. Бази даних NoSQL, використовуючи неструктурований підхід, пропонують багато ефективних способів обробки даних.

В даній дипломній роботі однією з головних вимог до бази даних є швидкість доступу та масштабованість, тому була обрана нереляційна база даних.

Є декілька типів нереляційних баз даних, які визначаються системою зберігання даних. Розрізняють наступні моделі баз даних: ключ-значення, документно-орієнтована та графова.

Порівнявши легкість доступу до даних в усіх трьох моделях, обрано документно-орієнтовану модель бази даних, а саме Cloud Firestore.

Cloud Firestore — це гнучка, масштабована база даних для мобільних, веб- та серверних розробок від Firebase та Google Cloud Platform, яка є наступним поколінням Firebase Realtime Database. Як і Firebase Realtime Database, вона дозволяє синхронізувати дані застосунків між клієнтами та пропонує офлайн підтримку для мобільних та веб-застосунків, що дозволяє створювати додатки, які будуть працювати незалежно від швидкості мережі чи наявності підключення до Інтернету.

Firebase — платформа для розробки веб- та мобільних застосунків від компанії Google. Станом на березень 2020 року має 19 сервісів, які використовуються в понад 1,5 млн застосунків.

Розглянемо деякі сервіси для збереження даних, що використовуються в розробленому мобільному застосунку.

Firebase Authentication — сервіс для автентифікації користувачів з використанням коду на стороні клієнта. За допомогою цієї служби розробник може налаштувати різні способи авторизації користувача в системі.

Firebase Storage — сервіс для безпечного завантаження та скачування файлів незалежно від якості мережі. Зазвичай використовується для збереження аудіо- та відеофайлів, зображень чи інших файлів, створених користувачами.

3.3. Обґрунтування вибору середовища розробки

Найпопулярнішими середовищами розробки для створення мобільних застосунків є Android Studio та Visual Studio Code.

Android Studio — інтегроване середовище розробки для роботи на платформі Android, побудоване на основі середовища розробки IntelliJ IDEA від компанії JetBrains. Android Studio надає засоби для створення емуляторів Android-пристроїв з різними характеристиками, тобто розробник може тестувати сумісність створеної програмної системи з різними версіями Android-платформи та вигляд застосунку на пристроях з різними діагоналями екранів.

Visual Studio Code — редактор вихідного коду для створення кросплатформних веб- та хмарних застосунків, розроблений компанією Microsoft для Windows, macOS, Linux. Редактор містить вбудований відладчик, інструменти для роботи з системою керування версіями (Git) та засоби рефакторингу. Також VS Code має багато можливостей для персоналізації, дозволяючи користувачам змінювати тему, комбінації клавіш, налаштування і встановлювати розширення, які надають додаткові функції.

Обидва середовища розробки містять всі необхідні інструменти для створення і налагодження Flutter-застосунків. Visual Studio Code є більш легким редактором

коду, що не сильно навантажує процесор і більш швидко перебудовує застосунок після деяких змін. Це і стало вирішальним фактором у виборі середовища розробки.

3.4. Висновки до розділу

У даному розділі описано мову програмування та технології, використані при розробці мобільного застосунку. Обрано фреймворк Flutter, оскільки він є більш швидким, має бібліотеку для легкого доступу до сенсорів мобільного телефону, велику кількість інструментів для розробки і тестування та містить докладний опис кожного віджета у документації. Середовищем розробки обрано Visual Studio Code, оскільки воно містить всі необхідні інструменти для створення і налагодження Flutter-застосунків і несильно навантажує процесор. В якості бази даних обрано нереляційну базу Cloud Firestore, оскільки вона надає швидкий доступ до збережених даних, а сервіс Firebase надає інструменти, що дозволяють зберігати дані в базу не одразу, а коли з’явиться інтернет на мобільному пристрої.

4. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

4.1. Архітектура системи

Розроблений мобільний застосунок може виявляти та ідентифікувати нетипову поведінку при керуванні транспортним засобом згідно з даними сенсорів мобільного телефону. Система складається з двох частин: офлайн частини — Моделювання поведінки водія та онлайн частини — Аналіз поведінки водія (рисунок 4.1).

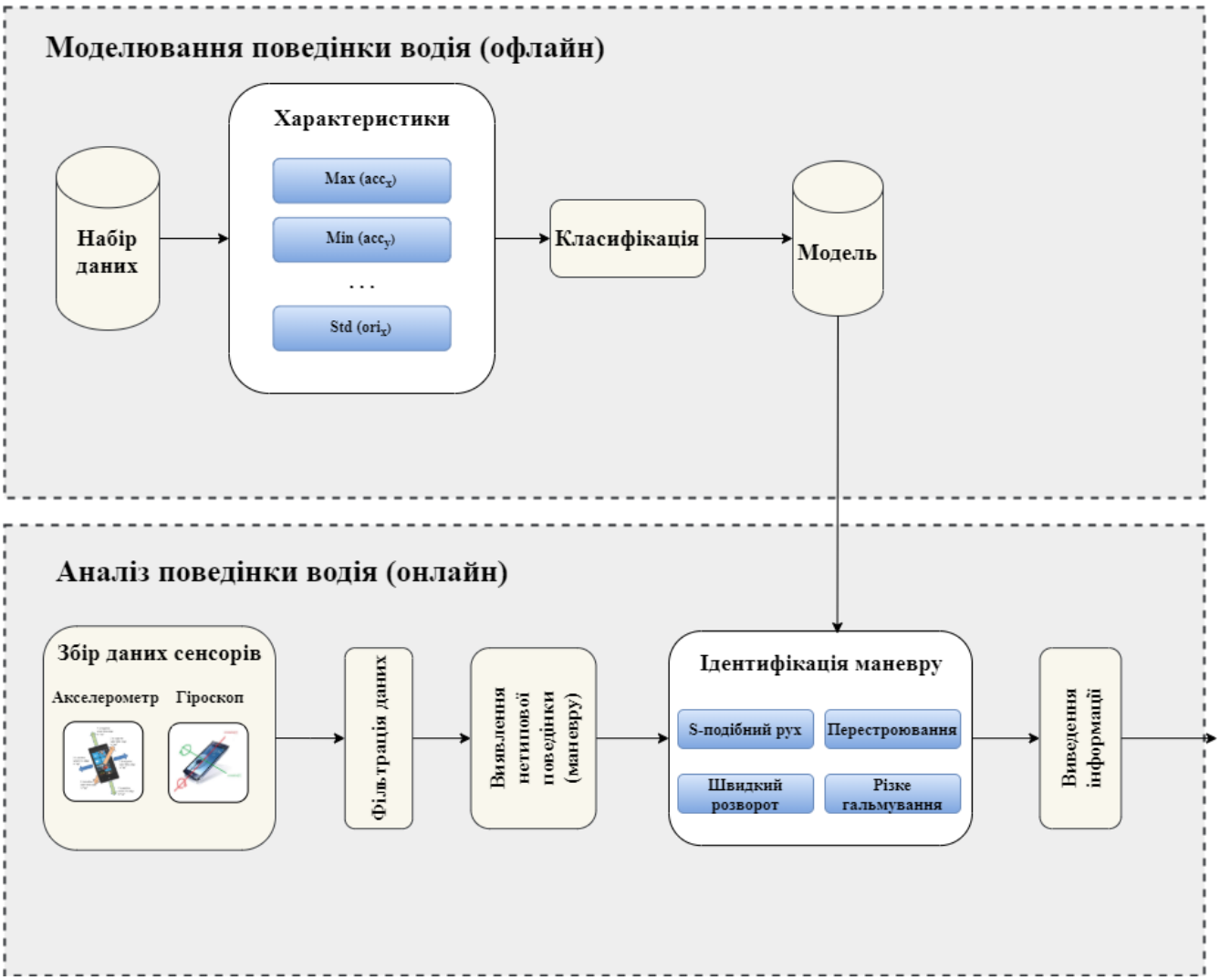


Рисунок 4.1 — Архітектура мобільного застосунку

В офлайн частині система класифікує різні види нетипової поведінки людини при керуванні транспортним засобом на основі даних тренувальних поїздок.

Використовуючи дані сенсорів кожної поїздки, виокремлено 16 характеристик для ідентифікації кожного виду нетипової поведінки. Модель — це набір об’єктів, що містять в собі 16 характеристик і клас нетипової поведінки.

Онлайн частина, Аналіз поведінки водія, записує дані акселерометра і гіроскопа. Після закінчення поїздки ці дані аналізуються та виявляється початок і кінець всіх маневрів. Після цього, в блоці Ідентифікація маневру система отримує дані з моделі і, використовуючи алгоритм KNN, класифікує дану нетипову поведінку та виводить інформацію про поїздку користувачу.

4.2. Діаграма прецедентів

Діаграма прецедентів застосовується для моделювання виду системи з точки зору зовнішнього спостерігача.

Актор — це будь-яка сутність, що взаємодіє з розробленою системою ззовні. Актор може бути людиною, іншою системою, підсистемою або класом [15].

Прецедент — це опис набору послідовних подій, що можуть виконуватись Актором в системі.

Для опису взаємодії акторів та прецедентів на діаграмі відображаються відносини.

Стандартні види відносин між акторами і подіями:

- асоціації (association);
- розширення (extend);
- узагальнення (generalization);
- включення (include).

Діаграма прецедентів розробленого застосунку зображена на рисунку 4.2.

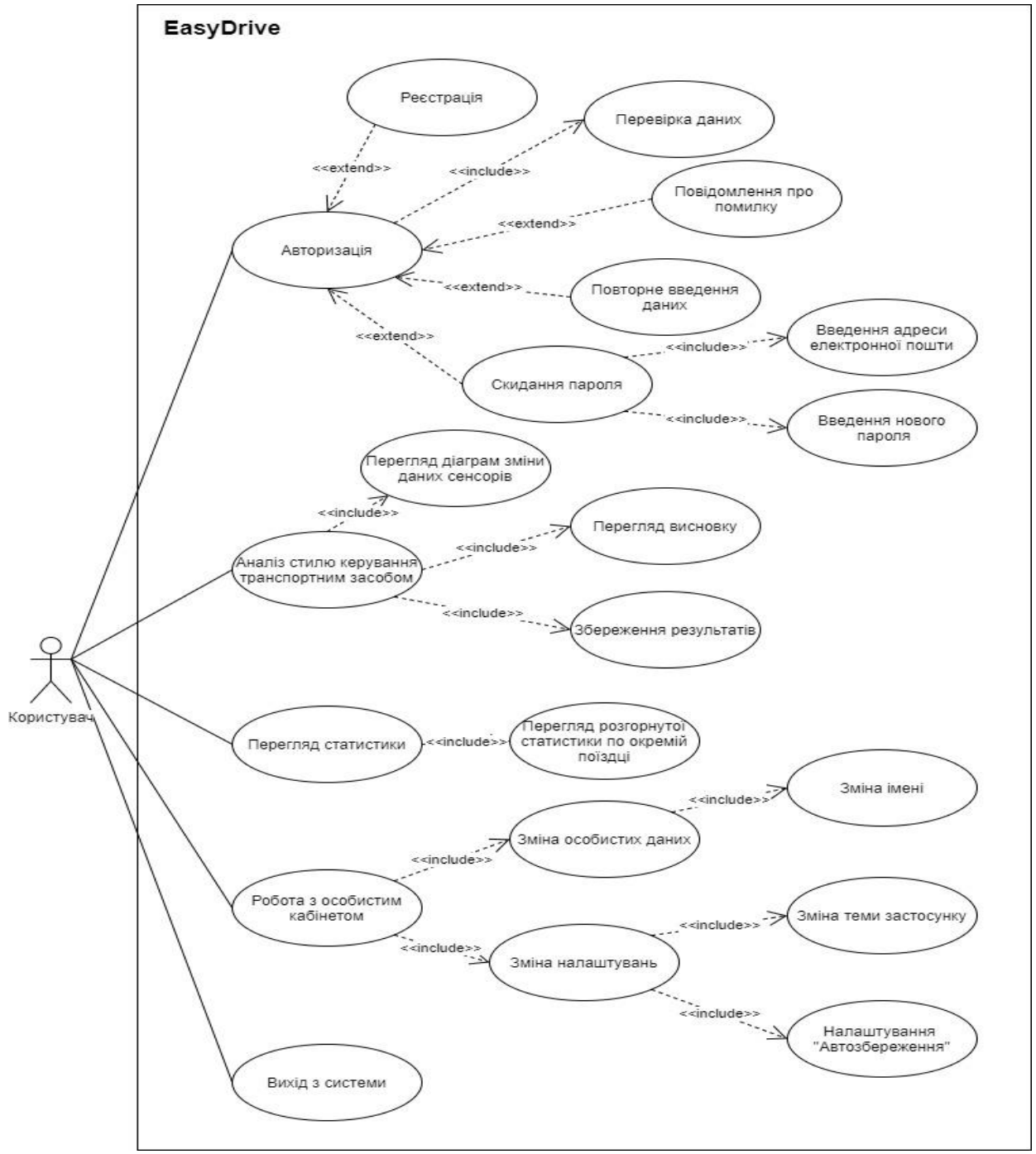


Рисунок 4.2 — Діаграма прецедентів

Головним актором є Користувач, який може авторизуватися/зареєструватися. Після входу в систему користувачу доступний наступний функціонал: аналіз стилю керування транспортним засобом, що включає перегляд діаграм зміни даних сенсорів, збереження поїздки та перегляд висновку щодо поведінки; перегляд загальної статистики та розгорнутої статистики окремої поїздки; редагування особистих даних, зміна налаштувань застосунку та вихід з системи.

4.3. Діаграма класів

Діаграма класів — це діаграма призначена для представлення моделі статичної структури програмної системи. На діаграмі клас зображується прямокутником, розділеним на 3 частини: ім’я класу, атрибути та операції. При розробці системи важливу роль відіграють як структура класу, так і відносини між ними. Базові типи зв’язків: асоціація, агрегація, композиція і узагальнення.

Класи розробленої програми і зв’язки між ними зображені на рисунку 4.3.

При програмуванні мобільного застосунку використана технологія Flutter. В цій технології все побудовано на віджетах: віджет без стану (StatelessWidget) і віджет зі станом (StatefulWidget). Користувацькі класи наслідують ці віджети. Основні класи програми:

- Клас MyApp — містить функцію main(), що є точкою входу в програму.
- Клас MyAuth — клас, що містить функції для реєстрації/авторизації в системі.
- SignInSignUpPage — StatefulWidget, що є представленням сторінки реєстрації та авторизації і містить основні методи валідації форми.
- HomePage — клас зі станом, що є головною сторінкою.
- SensorsPage — клас, що наслідується від StatefulWidget і містить методи для збереження поїздки, відображення динамічних графіків, аналізу поїздки і поведінки водія.
- TripAnalysPage — клас, що є представленням сторінки аналізу поїздки.
- StatisticPage — клас, що містить методи доступу до бази даних і відображення статистики користувача.
- ProfilePage — клас, що наслідує віджет зі станом; містить методи доступу до бази даних, вибору і завантаження зображення профілю, зміни особистих даних та налаштувань.

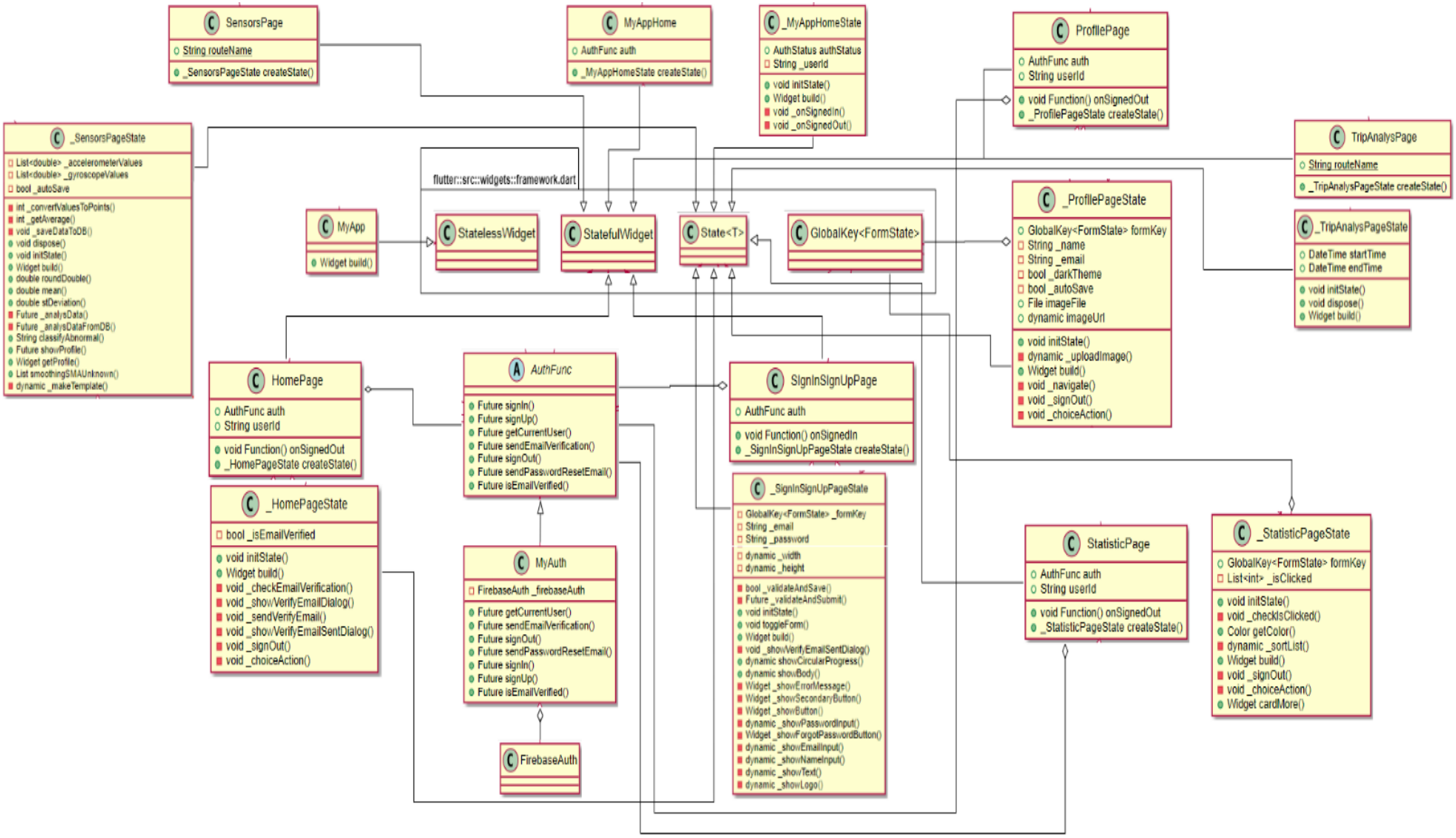


Рисунок 4.3 — Діаграма класів

4.4. Діаграма розгортання

Діаграма розгортання (Deployment diagram) призначена для візуалізації елементів та компонентів програми, які існують лише на етапі її виконання. Основними елементами даної діаграми є вузли, компоненти і зв’язки між ними. Вузол — деякий фізичний елемент системи (комп’ютер, датчик, модем, мобільний телефон і т. д.). Каналом зв'язку між вузлами виступає фізичне з'єднання (наприклад, оптоволоконна лінія або супутниковий зв'язок).

Діаграма розгортання на Андроїд-пристрої розробленого мобільного застосунку зображена на рисунку 4.4.

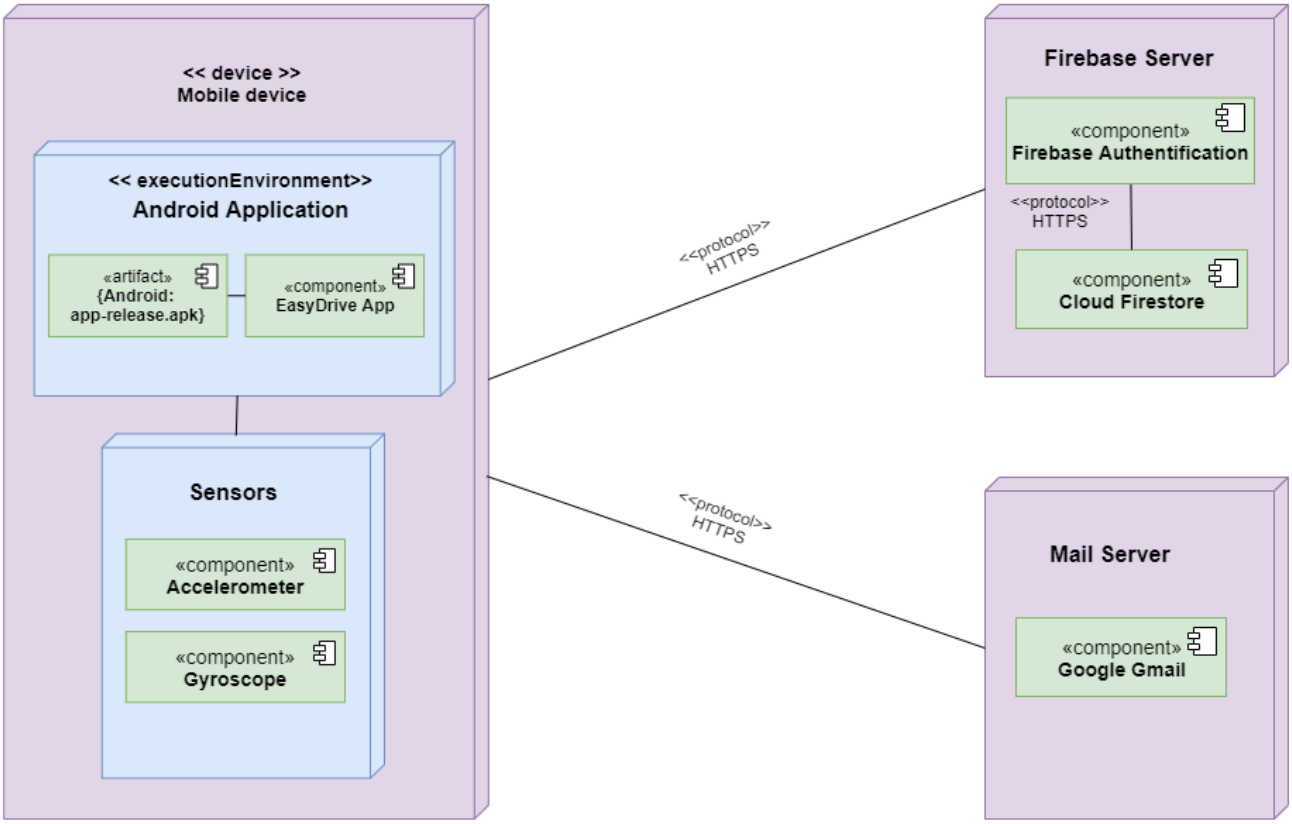


Рисунок 4.4 — Діаграма розгортання

На рисунку 4.4 зображено 3 основних вузли: мобільний телефон, сервер бази даних (Firebase Server), поштовий сервер та їх основні компоненти. Середовищем виконання є система Андроїд. Для виконання необхідно мати файл “app-release.apk”, використовуючи який можна встановити розроблену систему “EasyDrive”. Як можна побачити на діаграмі, застосунок використовує сенсори мобільного телефону.

З’єднання з сервером бази даних та поштовим сервером відбувається з використанням протоколу HTTPS — безпечним протоколом передачі даних в/з мережі, що шифрує дані, використовуючи криптографічні протоколи SSL та TLS.

4.5. Опис структури бази даних

Розроблений мобільний застосунок використовує нереляційну базу даних Cloud Firestore для збереження інформації про користувачів, даних поїздок і показників сенсорів під час керування транспортним засобом. В Cloud Firestore для збереження даних використовуються колекції та документи. Документ — це запис, який містить будь-які поля. Документи об’єднуються в колекції [16]. Якщо проводити аналогію з реляційною базою, то колекція — це таблиця, а документ — це запис в цій таблиці.

Структура бази даних мобільного застосунку наведена на рисунку 4.5.

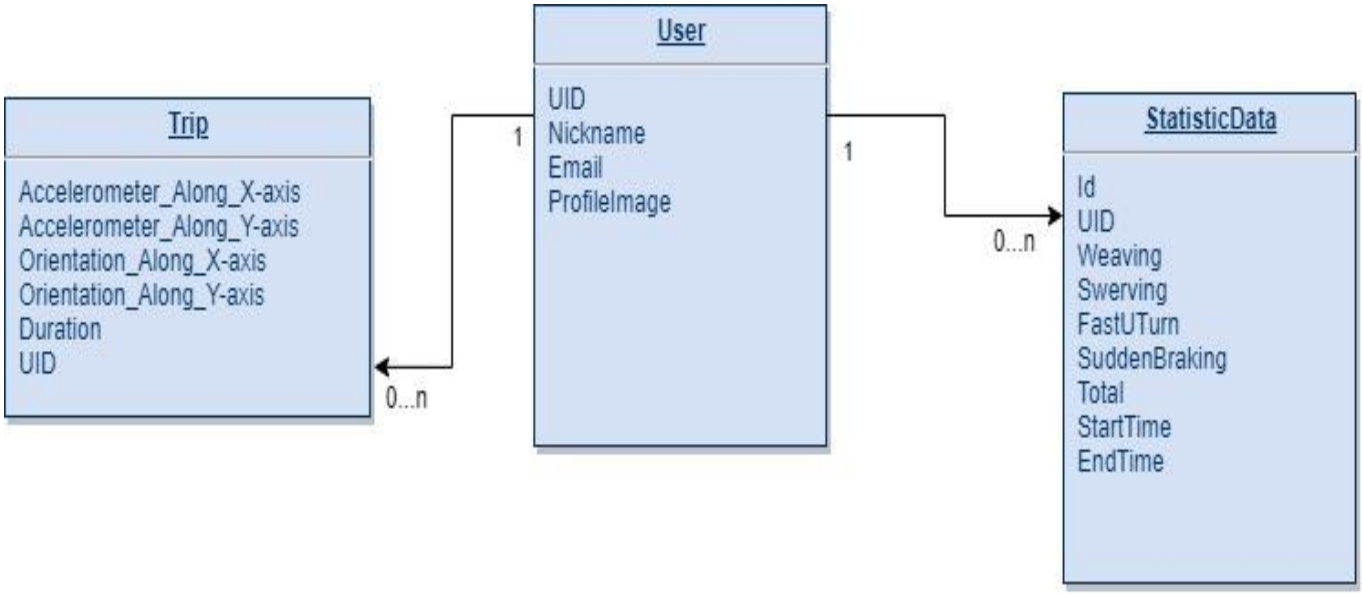


Рисунок 4.5 — Структура бази даних

У колекції “Trip” зберігаються записи з даними сенсорів кожної тренувальної поїздки. Колекція “User” містить документи з даними користувачів системи. Колекція “StatisticData” зберігає аналітичні дані кожної поїздки (початок, кінець поїздки та кількість різних видів нетипової поведінки).

4.6. Алгоритм аналізу поїздки

Після натискання користувачем кнопки “Старт” на головній сторінці починається відстеження поїздки. Кожні півсекунди в масив зберігається об’єкт, що містить дані сенсорів (акселерометра та гіроскопа): acc_x, acc_y, ori_x, ori_y.

Коли користувач натискає кнопку “Стоп” запис поїздки припиняється і починається аналіз збережених даних. Збережені дані фільтруються методом рухомого середнього з коефіцієнтом 2. Потім стандартне відхилення $\sigma_{acc,x}$ двох сусідніх об’єктів масиву згладжених даних порівнюється з пороговим значенням. Якщо це значення більше за порогове значення, то система виявляє, що розпочався небезпечний маневр.

Коли показники стандартного відхилення стають меншими за порогове значення, система розпізнає, що нетипова поведінка при керуванні транспортним засобом завершилась. Тоді запускається функція для класифікації небезпечного маневру, яка в якості параметрів отримує індекси початку та кінця нетипової поведінки.

Далі, розраховуються значення 16 атрибутів, необхідних для ідентифікації нетипової поведінки. Використовуючи алгоритм k-найближчих сусідів (KNN), визначається клас небезпечного маневру. Назва класу повертається в головну функцію і відповідна змінна, що зберігає кількість даного небезпечного маневру, збільшується на 1. Потім аналіз даних поїздки продовжується.

Блок-схема вищезазначеного алгоритму наведена на рисунку 4.6.

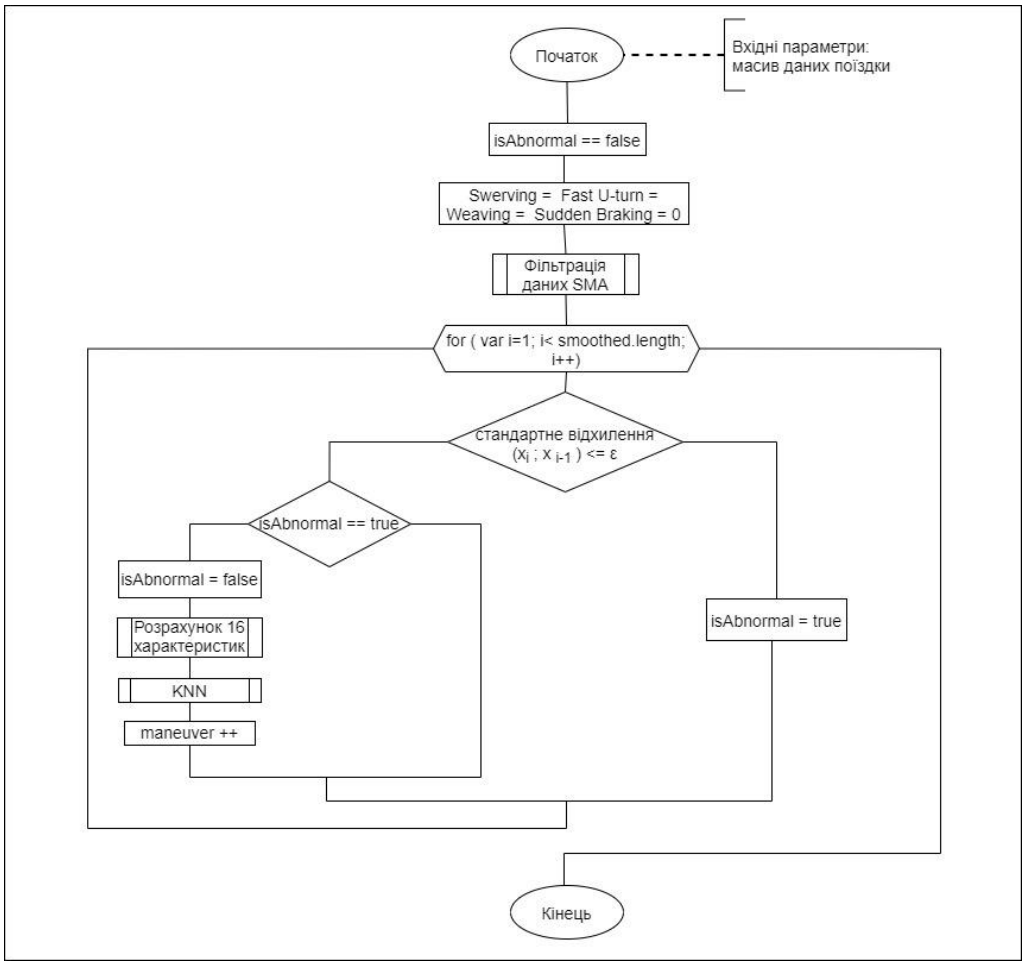


Рисунок 4.6 — Блок-схема алгоритму аналізу поїздки

Після аналізу всієї поїздки користувачу відображається, яку кількість кожного виду нетипової поведінки він здійснив.

4.7. Алгоритм розпізнавання поведінки водія

Поведінка водія — один з багатьох ключових факторів, який необхідно роздивитися для підвищення безпеки дорожнього руху. Основна складність визначення поведінки полягає у відсутності єдиного стандартного методу її моделювання, проте можна виділити дві основні концепції досліджень: кількісні та якісні. Якісний підхід спрямований на пояснення взаємозв'язку між стилями керування і суб'єктивними змінними, такими як людські фактори, особистість, емоції,

тоді як кількісний підхід фокусується на об’єктивних змінних, які можуть бути розраховані електронними пристроями, наприклад, швидкість, прискорення і т. д.

Запропонований алгоритм профілювання поведінки водія, розроблений у дипломній роботі, враховує небезпечні маневри, що можуть призвести до дорожньо-транспортної пригоди. Для розпізнавання поведінки водія при керуванні транспортним засобом використовується індекс безпеки водія (driver safety index, SI) [17]. Вся подорож ділиться на інтервали по 1 хвилині, і на кожному з цих інтервалів знаходиться індекс безпеки за формулою 4.1:

$$SI = 1 - \{n_w\beta_w + n_s\beta_s + n_F\beta_F + n_{SB}\beta_{SB}\}, \quad (4.1)$$

де n_w – кількість S-подібних рухів;

n_S – кількість перестроювань;

n_F – кількість швидких розворотів;

n_{SB} –кількість раптових гальмувань;

β_w – “вага” S-подібного руху;

β_S – “вага” перестроювання;

β_F – “вага” швидкого розвороту;

β_{SB} – “вага” раптового гальмування.

Для всіх небезпечних маневрів прийнято $\beta = 0.25$.

Після знаходження індексу безпеки на кожному з інтервалів, обчислюється середній індекс безпеки всього маршруту, за яким визначається тип водія.

У даній роботі визначено 4 типи поведінки: дуже безпечна, безпечна, агресивна і дуже агресивна. В таблиці 3.3 вказані індекси безпеки для кожної поведінки.

Таблиця 3.3 Індекси безпеки для різних типів поведінки

Тип поведінки	Індекс безпеки, SI
Дуже безпечний	0.9 — 1
Безпечний	0.7 — 0.89
Агресивний	0.5 — 0.69
Дуже агресивний	< 0.5

Тобто, чим більший індекс, тим менша кількість маневрів була здійснена під час поїздки і тим безпечніша була поведінка водія транспортного засобу.

4.8. Висновки до розділу

У даному розділі наведено архітектуру мобільного застосунку, діаграму класів з описом основних класів системи, діаграму прецедентів, діаграму розгортання і структуру бази даних.

Також даний розділ містить блок-схему і пояснення алгоритму аналізу поїздки. Крім цього, тут наведено алгоритм розпізнавання поведінки водія.

5. МЕТОДИКА РОБОТИ КОРИСТУВАЧА З МОБІЛЬНИМ ЗАСТОСУНКОМ

Для точного розпізнавання та ідентифікації нетипової поведінки при керуванні транспортним засобом даним мобільним застосунком необхідно дотримуватися вимог при його встановленні та використанні.

5.1. Системні вимоги

Розроблений мобільний застосунок призначений для використання на Android та iOS пристроях. Для встановлення застосунку на Android, пристрій повинен мати не менше 20 МБ вільної пам’яті і версію операційної системи Android Jelly Bean (4.1.x) чи вище. Для встановлення застосунку на пристрій під керуванням iOS версія операційної системи має бути 8 і вище.

5.2. Встановлення мобільного застосунку

Для встановлення мобільного застосунку на iOS пристрій необхідно завантажити файл app_release.ipa, потім запустити його і натиснути кнопку “Встановити” (“Install”). Для інсталяції застосунку на пристрій на платформі Android треба завантажити файл app_release.apk, запустити його і почати встановлення, натиснувши на кнопку “Встановити” (“Install”) (рисунок 5.1).

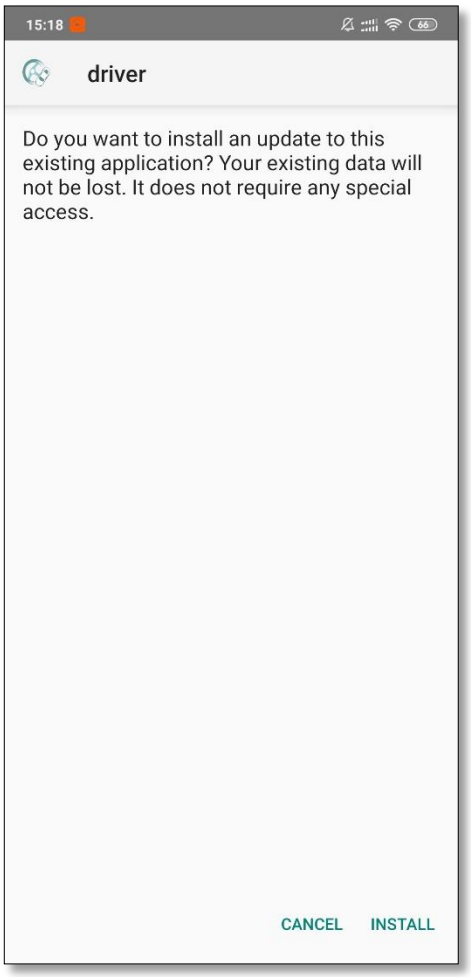


Рисунок 5.1 — Інсталяція застосунку на Android пристрій

5.3. Запуск і варіанти використання застосунку

Для запуску застосунку необхідно знайти на робочому столі іконку застосунку і натиснути на неї (рисунок 5.2).

Рисунок 5.2 — Іконка мобільного застосунку

Після натискання на іконку починається запуск застосунку (рисунок 5.3).

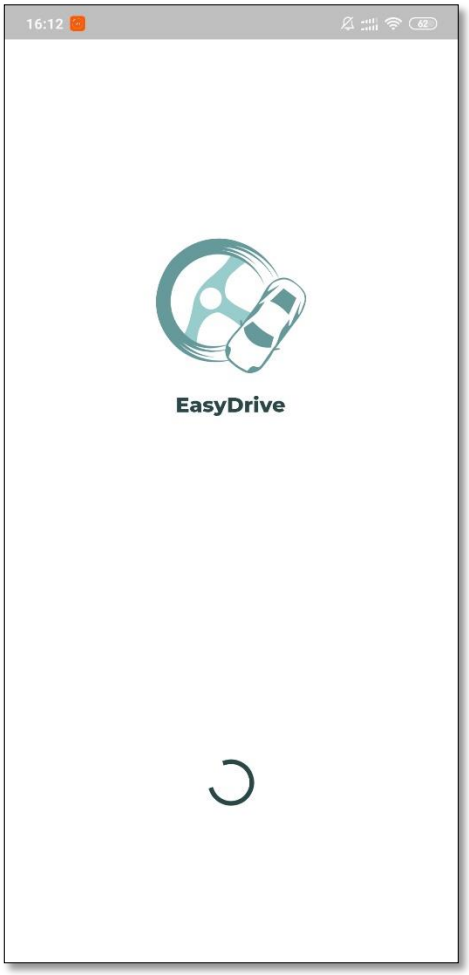
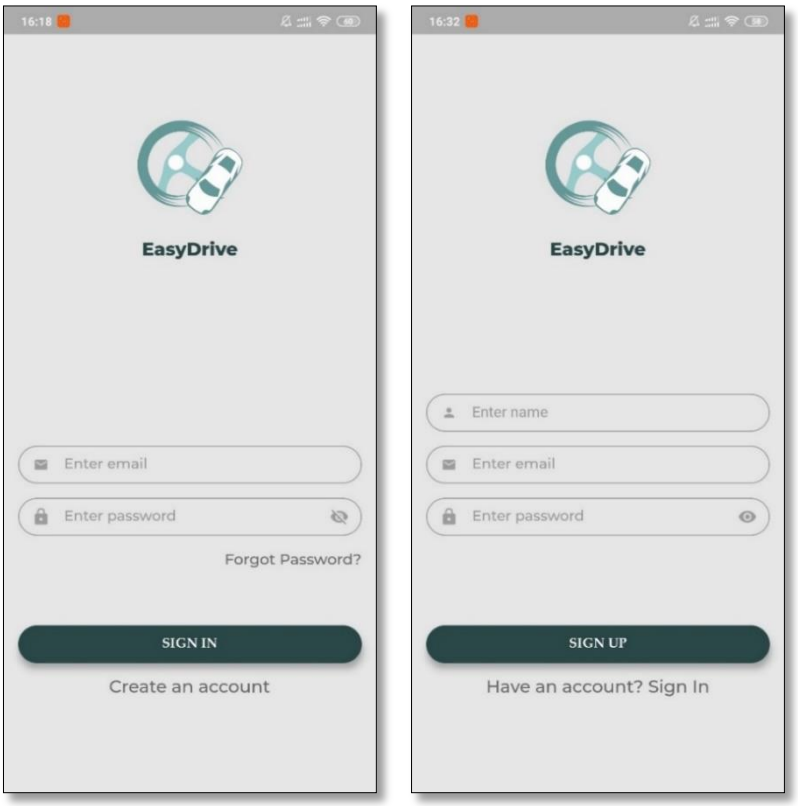


Рисунок 5.3 — Запуск застосунку

При першому запуску застосунку відкривається сторінка входу в систему. Для авторизації необхідно ввести адресу електронної пошти та пароль і натиснути кнопку “Увійти” (“Sign in”) (рисунок 5.4а).

Якщо користувач не зареєстрований в системі, необхідно натиснути на кнопку “Створити обліковий запис” (“Create an account”) і відкриється сторінка реєстрації (рисунок 5.4б).

Для реєстрації користувачу необхідно ввести ім’я, електронну пошту та пароль. Якщо в поля вводу введені невірні дані, виведеться повідомлення про помилку (рисунок 5.5).



а) б)

Рисунок 5.4 — Інтерфейс застосунку: а– Сторінка входу в систему, б — Сторінка реєстрації в системі

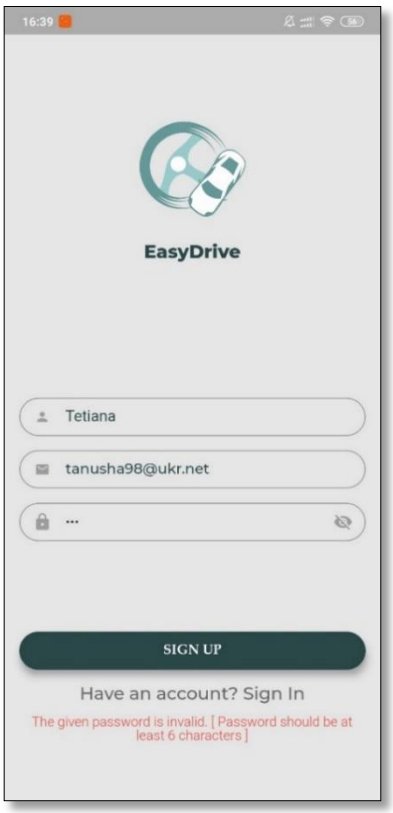


Рисунок 5.5 — Повідомлення про неправильний пароль

Коли користувач введе всі дані вірно і натисне кнопку “Зареєструватися” (“Sign up”), на екрані відобразиться вікно з повідомленням про те, що на вказану електронну пошту надіслано лист для підтвердження реєстрації (рисунок 5.6).

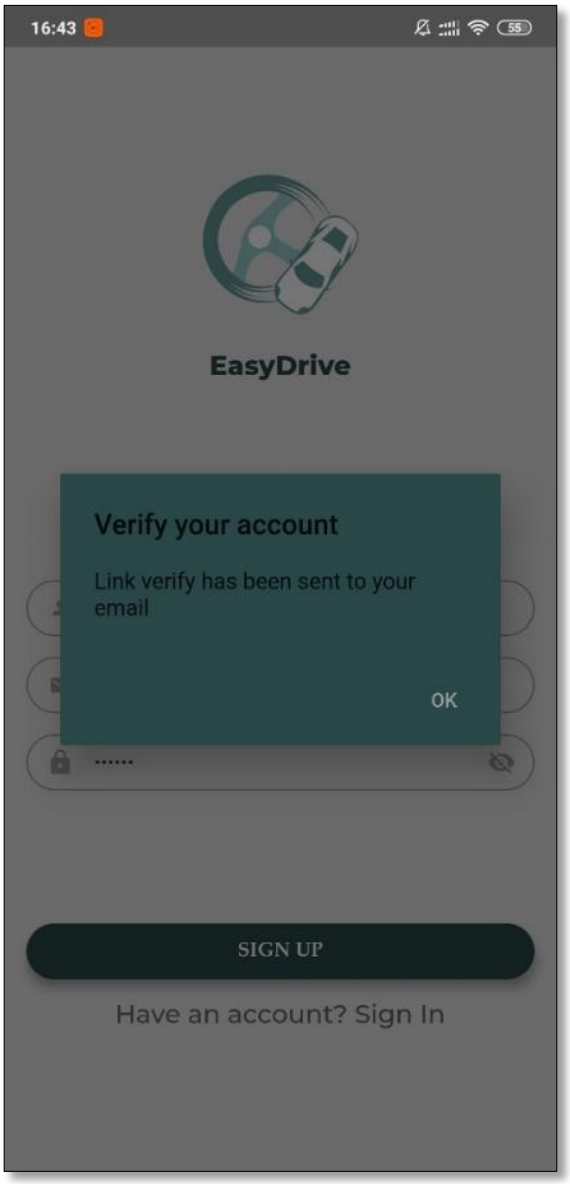


Рисунок 5.6 — Повідомлення з проханням підтвердити реєстрацію

Після натискання кнопки “ОК” користувач повертається на сторінку входу в систему, де він має ввести адресу електронної пошти та пароль.

Якщо користувач невірно ввів дані, буде відображено повідомлення про помилку (рисунок 5.7).

Рисунок 5.7 — Відображення повідомлення про невірний пароль

Як видно на рисунку 5.7, пароль не відображається, кожен символ замінюється крапками. Для того, щоб побачити текст паролю необхідно натиснути на іконку “ока” у полі вводу пароля (рисунок 5.8).

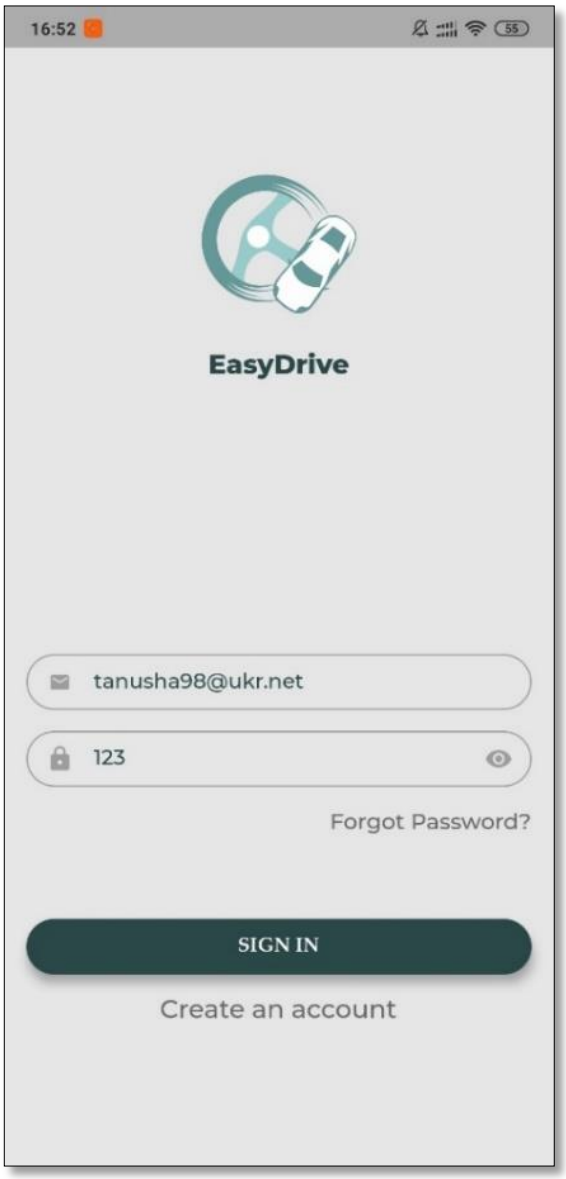


Рисунок 5.8 — Відображення паролю

Якщо користувач забув пароль, треба натиснути на кнопку “Забули пароль?” (“Forgot Password?”). Тоді відкриється сторінка скидання пароля, де необхідно вказати електронну пошту, на яку система має відправити лист з посиланням і натиснути кнопку “Відправити” (“Submit”) (рисунок 5.9).

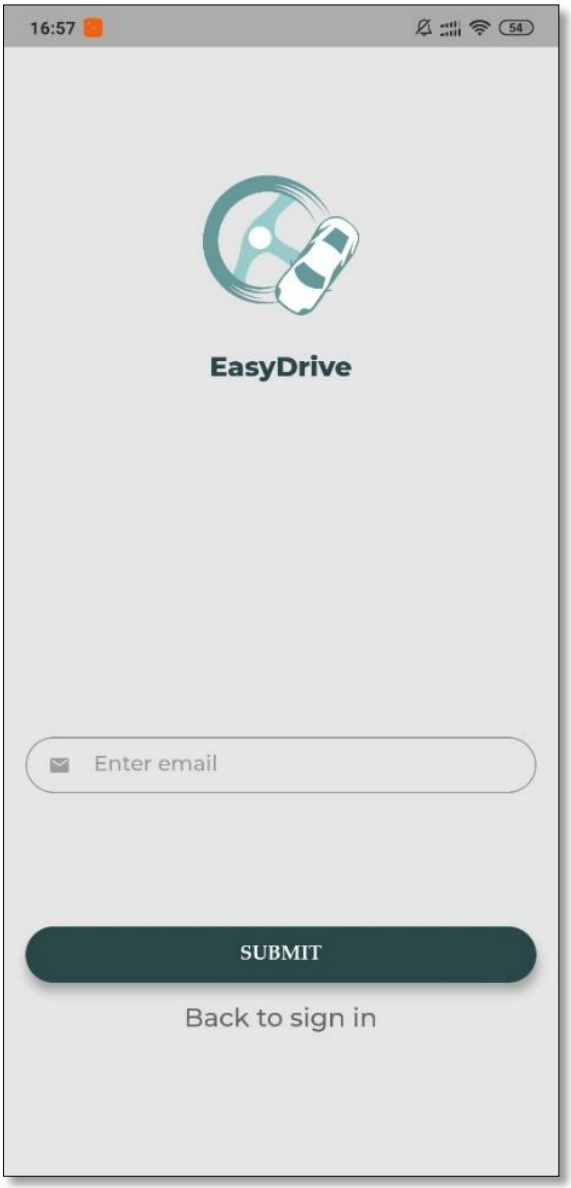
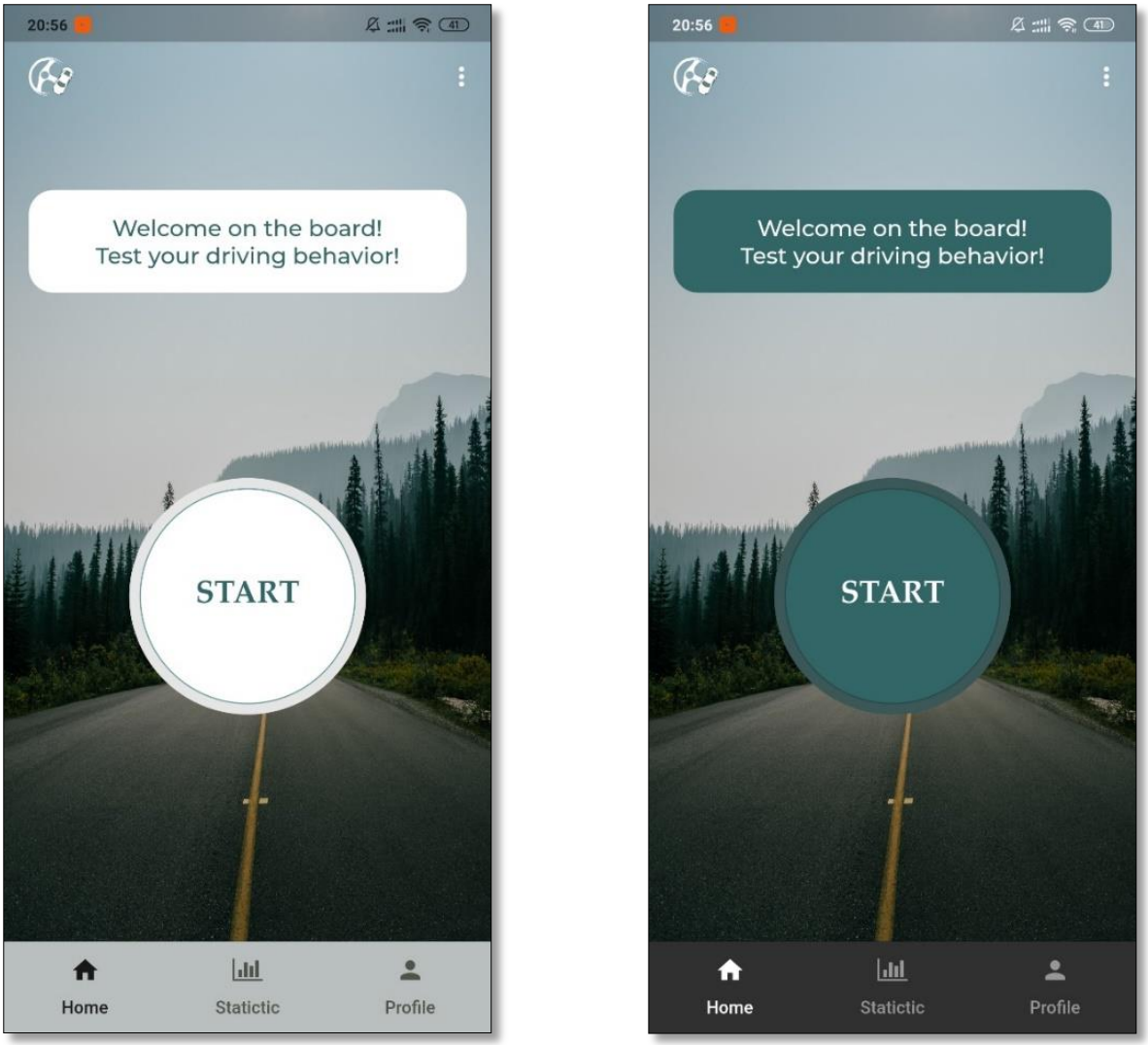


Рисунок 5.9 — Сторінка скидання пароля

Якщо користувач на сторінці входу в систему (рисунок 5.4а) ввів вірні дані — відкривається головна сторінка з привітанням та кнопкою “Старт” (“Start”) (рисунок 5.10). За замовчуванням застосунок має світлу тему (рисунок 5.10а), яку можна змінити на темну на сторінці “Профіль” (“Profile”) (рисунок 5.10б).



а) б)

Рисунок 5.10 — Головна сторінка застосунку: а — Світла тема, б — Темна тема

Для запису і аналізу поїздки необхідно на головній сторінці натиснути кнопку “Старт” (“Start”). Після цього відкриється сторінка поїздки, на якій відображаються динамічні графіки даних акселерометра та гіроскопа (рисунок 5.11).

Для того, щоб повернутися на головну сторінку без збереження даних поїздки, необхідно натиснути на стрілку в лівому верхньому куті. Після цього відкриється діалогове вікно з запитанням чи хочете ви вийти без збереження даних поїздки (рисунок 5.12).

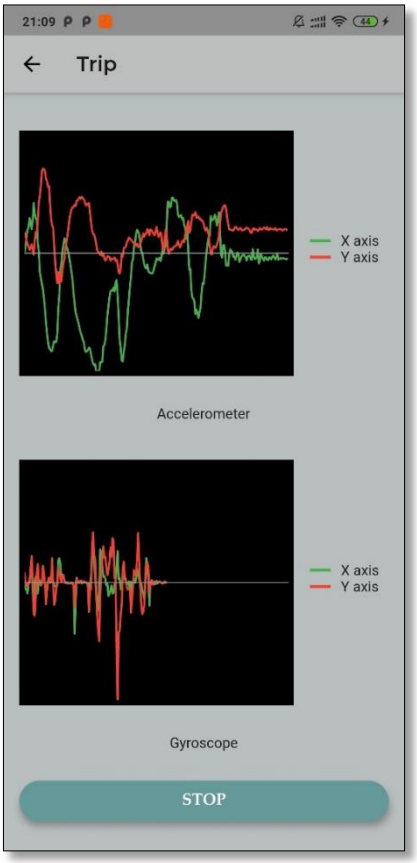


Рисунок 5.11 — Сторінка поїздки в реальному часі

Рисунок 5.12 — Діалогове вікно виходу із запису поїздки

Для того, щоб побачити аналіз поїздки потрібно натиснути кнопку “Стоп” (“Stop”). Якщо встановлений режим автозбереження (за замовчуванням значення автозбереження — хиба, його можна змінити на сторінці “Профіль”), то дані поїздки зберігаються в базі. Якщо режим автозбереження не встановлений, відобразиться діалогове вікно із запитанням чи хочете ви зберегти дані поїздки (рисунок 5.13).

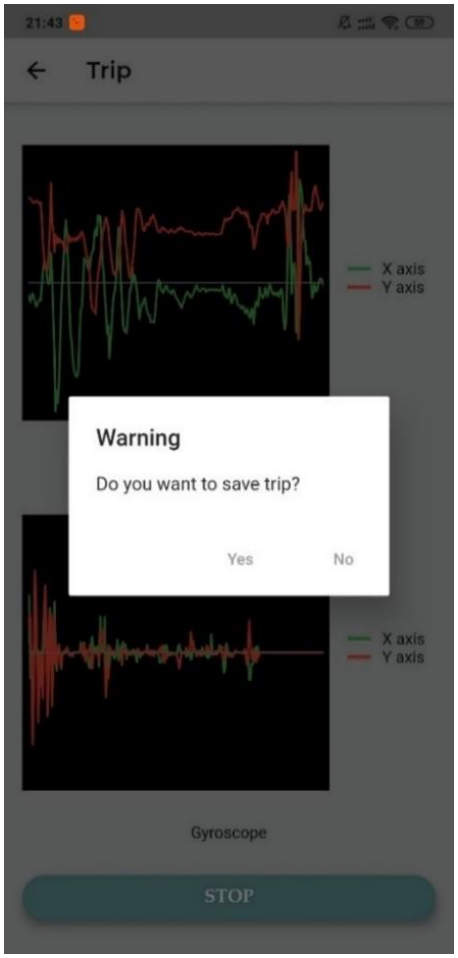


Рисунок 5.13 — Діалогове вікно припинення запису поїздки

Після вибору зберегти поїздку чи ні відкриється повідомлення з описом поведінки водія в даній поїздці (рисунок 5.14).

Коли користувач закриє повідомлення, відкриється сторінка з даними поїздки (кількістю небезпечних маневрів) (рисунок 5.15).

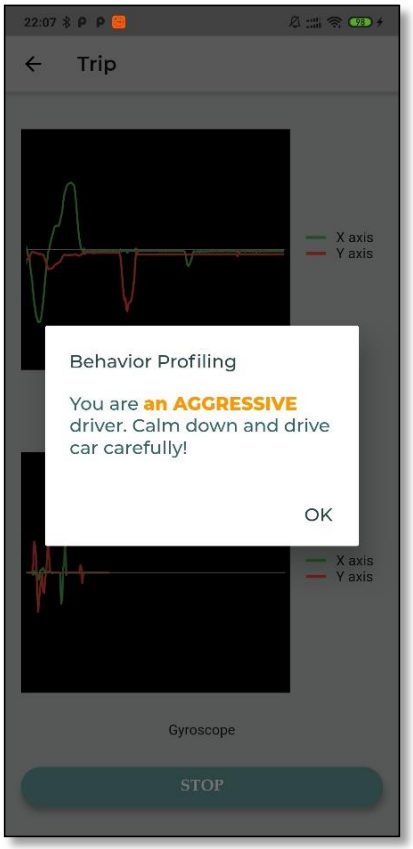


Рисунок 5.14 — Повідомлення з описом поведінки водія

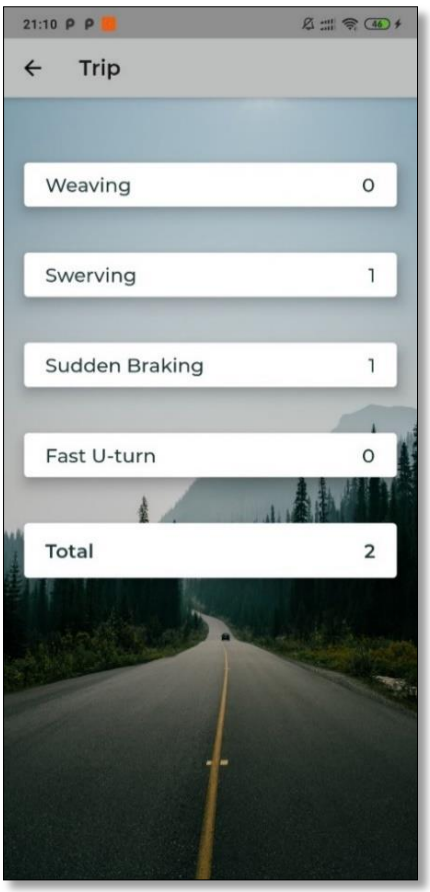
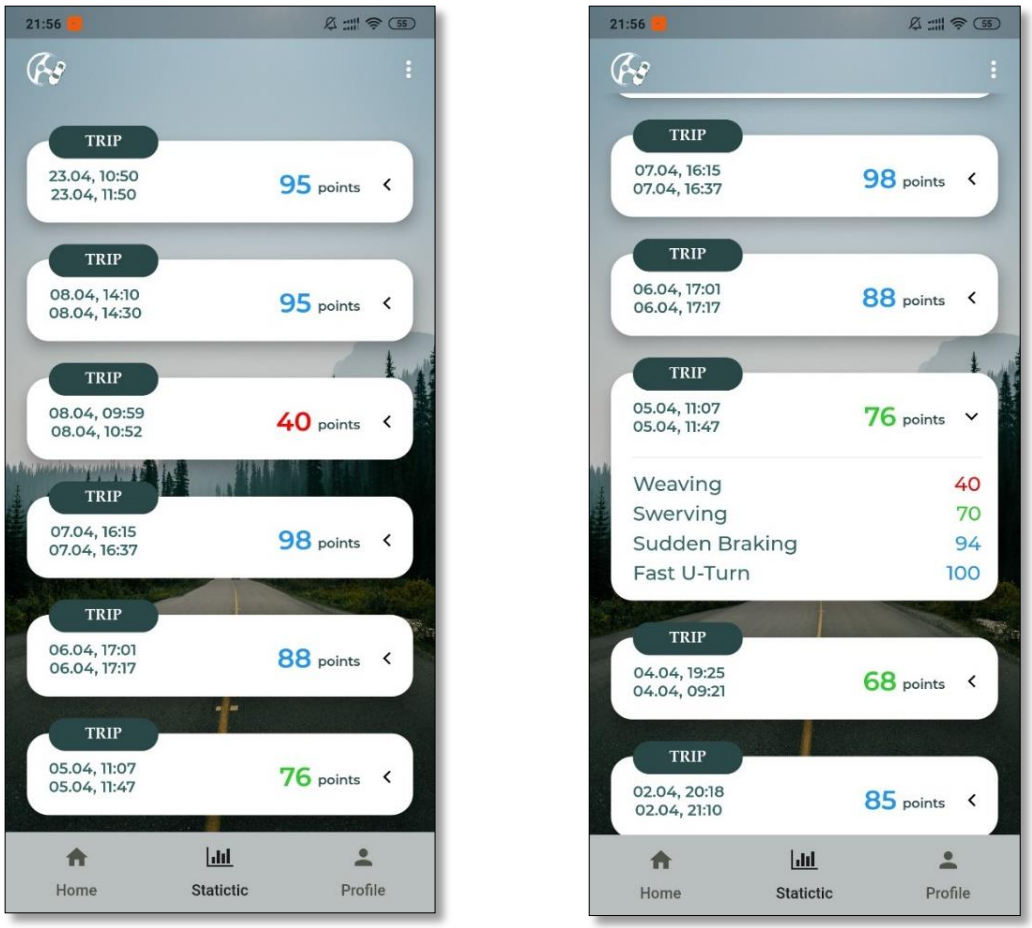


Рисунок 5.15 — Сторінка з даними поїздки

Для того, щоб переглянути статистику всіх поїздок необхідно перейти на сторінку “Статистика” (“Statistic”) (рисунок 5.16а). Щоб побачити розгорнуту статистику конкретної поїздки потрібно знайти її в переліку поїздок і натиснути на неї (рисунок 5.16б). Тоді, крім даних, що показуються в згорнутому вигляді, відобразяться бали за кожен з небезпечних маневрів. Чим більший бал, тим меншу кількість разів був здійснений даний маневр.



а) б)

Рисунок 5.16 — Сторінка статистики: а — статистика багатьох поїздок, б — розгорнута статистика вибраної поїздки

Для зміни налаштувань застосунку та персональних даних необхідно перейти на сторінку “Профіль” (“Profile”) (рисунок 5.17).

На даній сторінці можна встановити темну тему та режим автозбереження, змінивши положення перемикача. Також можна встановити/змінити фото профілю користувача. Для того, щоб вибрати фото з галереї необхідно натиснути на велике коло посередині (якщо користувач ще не встановлював фото профілю всередині кола

буде іконка людини, а якщо вже обирав — то вибране попереднє фото). Для фотографування зараз і встановлення цього фото потрібно натиснути на коло з камерою.

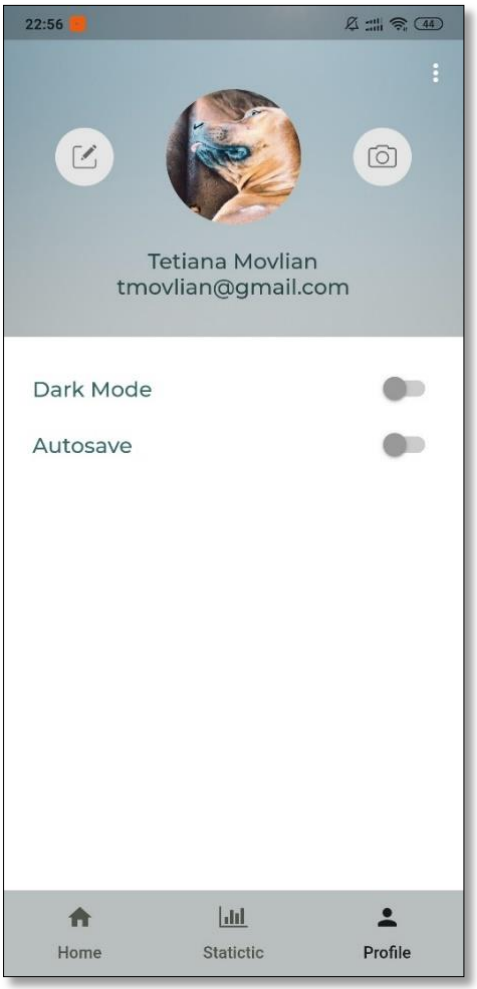


Рисунок 5.17 — Сторінка профілю користувача

Для зміни імені чи адреси електронної пошти треба натиснути на маленьке коло з олівцем. Після цього відкриється сторінка редагування (рисунок 5.18).

Для зміни даних потрібно ввести нові дані і натиснути кнопку “Змінити” (“Change”). Якщо введені дані невірні — виведеться повідомлення про помилку (рисунок 5.19).

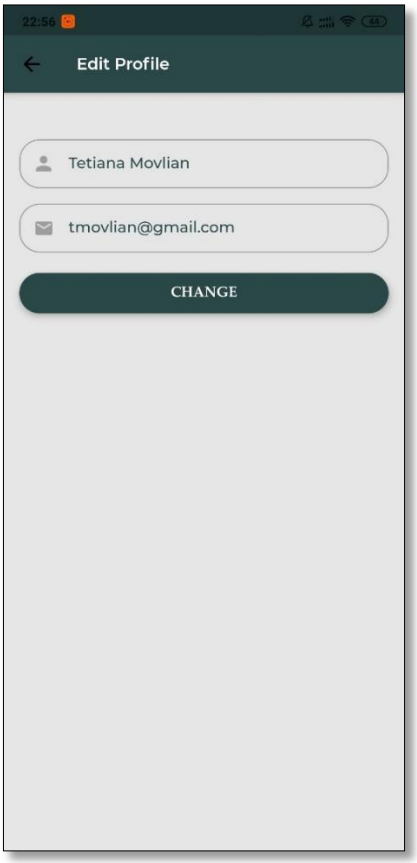


Рисунок 5.18 — Сторінка редагування даних користувача

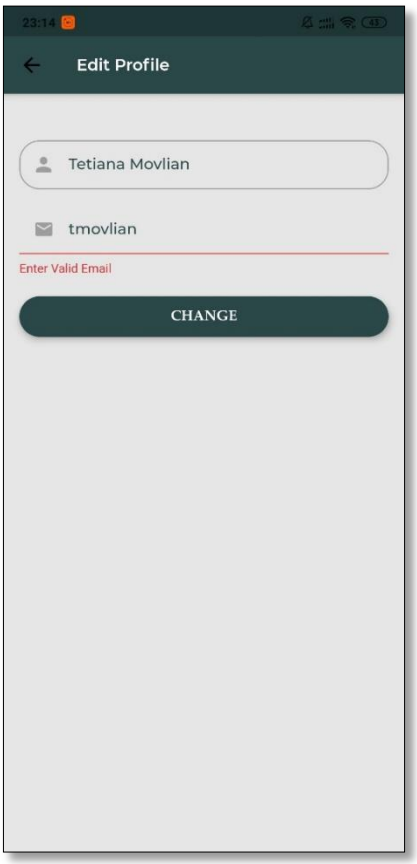


Рисунок 5.19 — Повідомлення про введення невірної email-адреси

Для виходу зі сторінки редагування без зміни даних потрібно натиснути на стрілку у верхньому лівому куті.

Для виходу з облікового запису можна натиснути на три крапки у верхньому правому куті на головній сторінці, сторінці перегляду статистики чи сторінці “Профіль”.

5.4. Висновки до розділу

У даному розділі описані системні вимоги до встановлення мобільного застосунку. Також наведено керівництво користувача, де описані всі варіанти використання розробленої програмної системи.

ВИСНОВКИ

В ході виконання роботи розглянута проблема виявлення і ідентифікації нетипової поведінки для підвищення безпеки керування транспортним засобом.

1. Проаналізовано існуючі мобільні застосунки, що відстежують поведінку водія для попередження ДТП або пом’якшення їх наслідків. Перевагами розглянутих застосунків є точність визначення небезпечної поведінки та відображення попереджень у реальному часі. Недоліками є те, що більшість розглянутих застосунків доступні лише на iOS-пристроях; багато функцій є платними; більшість застосунків записує і відображає відео в реальному часі, що займає багато місця в пам’яті телефону і витрачає багато заряду акумулятора.

2. Розроблено алгоритм розпізнавання поведінки водія при керуванні транспортним засобом, за яким виокремлено 4 типи поведінки: дуже безпечна, безпечна, агресивна та дуже агресивна. Тип поведінки визначається за допомогою індексу безпеки водія.

3. Проведено аналіз збережених “еталонних” поїздок та виокремлено характерні ознаки для класифікації небезпечних маневрів, до яких відносяться: діапазон значень, стандартне відхилення, середнє значення, максимальні та мінімальні значення, тривалість маневру.

4. Розроблено класифікаційну модель для ідентифікації нетипової поведінки, яка містить об’єкти, що зберігають 16 характеристик і клас небезпечного маневру.

5. Проаналізовано існуючі методи класифікації та фільтрації даних та виявлено, що найбільш точним методом фільтрації є метод рухомого ковзного середнього з коефіцієнтом 2, а найточнішим методом класифікації є KNN метод.

6. Розроблено нереляційну модель бази даних, що містить три колекції: для збереження особистих даних користувачів; для збереження показників сенсорів тренувальних поїздок та для збереження аналізу поїздок водіїв.

7. Проведено тестування програмного забезпечення на реальних даних поїздок.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Harrison B.L., Consolvo S., Choudhury . Using multi-modal sensing for human activity modeling in the real world / B.L.Harrison, S.Consolvo, T.Choudhury // Handbook of Ambient Intelligence and Smart Environments, Springer, 2010. — P. 463-478.

2. Preece S.J., Goulermas J.Y., Kenney L.P., Howard D. A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data. / S.J. Preece, J.Y. Goulermas, L.P. Kenney, D. Howard // IEEE Trans Biomed, Eng, 2010. — P. 871-879.

3. Jiadi Y., Yingying C., Xiangyu X. Sensing Vehicle Conditions for Detecting Driving Behaviors. / Y. Jiadi, C. Yingying, X. Xiangyu // SpringerBriefs in Electrical and Computer Engineering, Springer, 2010. — P. 75.

4. Stoichkov R. Android Smartphone Application for Driving Style Recognition. / R. Stoichkov // Multimodal Information Processing Group, Munich, 2013. — P. 3-7.

5. Hamdy A., Atia A., Sami M. Recognizing Driving Behavior and Road Anomaly using Smartphone Sensors [Електронний ресурс] / A. Hamdy, A. Atia, M. Sami. — 2017. — Режим доступу: https://www.researchgate.net /publication/ 318171110_ Recognizing _Driving_Behavior_and_Road_Anomaly_using_Smartphone_Sensors.

6. Fazeen M., Gozick B., Dantu R., Bhukhiya M., González M. Safe Driving Using Mobile Phones / M. Fazeen, B. Gozick, R. Dantu, M. Bhukhiya, M. González // IEEE Transactions on Intelligent Transportation Systems, 2012. — P. 7.

7. Derick J., Trivedi M. Driving Style Recognition Using a Smartphone as a Sensor Platform / J. Derick, M. Trivedi // 14th International IEEE Conference on Intelligent Transportation Systems, Washington, DC, USA, 2011. — P. 1609-1615.

8. Грешилов, А. А. Математические методы построения прогнозов / А. А. Грешилов, В. А. Стакун, А. А. Стакун. — Москва : Радио и связь, 1997. — 74 с.

9. Ferrer S., Ruiz T. S Travel behavior characterization using raw accelerometer data collected from smartphones. / S. Ferrer, T. Ruiz // Elsevier Ltd, Valencia, 2014. — P. 145-147.

10. Goldberger J., Roweis S., Hinton G., Salakhutdinov R. Neighbourhood Components Analysis [Електронний ресурс] / J. Goldberger, S. Roweis, G. Hinton, R. Salakhutdinov // Department of Computer Science, University of Toronto, 2004. — Режим доступу: http://papers.nips.cc/paper/2566-neighbourhood-components-analysis.pdf.

11. Zhongyang C., Jiadi Y., Yanmin Z., Yingying C., Minglu L. D3 : Abnormal Driving Behaviors Detection and Identification Using Smartphone Sensors. / C. Zhongyang, Y. Jiadi, Z. Yanmin, C. Yingying, L. Minglu // IEEE International Conference on Sensing, Communication, and Networking, 2015. — P. 5-6 .

12. Chen D., Cho K., Han S., Jin Z., Shin K. Invisible Sensing of Vehicle Steering with Smartphones / D. Chen, K. Cho, S. Han, Z. Jin, K. Shin // Department of Electrical Engineering and Computer Science University of Michigan, U.S.A, 2015 — P. 13 .

13. Windmill E. Flutter in action / E. Windmill //Manning Publications, Shelter Island, NY, 2020 — P. 341.

14. Buckett C. Dart in action / C. Buckett //Manning Publications, Shelter Island, NY, 2013 — P. 528.

15. Fowler M. UML Distilled. A Brief Guide to the Standard Object Modeling Language. / M. Fowler //Addison-Wesley, Pearson Education, Inc., 2004. — P.179.

16. Dominguez A., Bailey J., Djermanović D. Saving Data on Android: Learning Room, Firebase and SQLite with Kotlin / A. Dominguez, J. Bailey, D. Djermanović // Razeware LLC, 2019 — P. 306.

17. Saiprasert C., Thajchayapong S., Pholprasit T., Tanprasert C. Driver Behaviour Profiling using Smartphone Sensory Data in a V2I Environment / C. Saiprasert, S. Thajchayapong, T. Pholprasit, C. Tanprasert // Information Communication and Computing Research Unit National Electronics and Computer Technology Center, Thailand, 2016 — P. 6.

ДОДАТОК А

Інструментальні засоби розпізнавання поведінки людини з використанням сенсорів мобільного телефону

Специфікація

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ6291_20Б

Аркушів 2

Київ — 2020

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ6291_20Б	Записка.docx	Текстова частина дипломної роботи
Компоненти		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ6291_20Б 12-1	calculate.dart	Компонент, що містить розрахунки характеристик і метод KNN
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ6291_20Б 12-2	sensorsPage.dart	Компонент збереження поїздки, виводу інформації про небезпечні маневри і тип поведінки

ДОДАТОК Б

Інструментальні засоби розпізнавання поведінки людини з використанням сенсорів мобільного телефону

Текст програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ6291_20Б 12-1

Аркушів 13

Київ — 2020

calculate.dart

```
import 'dart:convert';
import 'dart:io';
import 'dart:math';

// Обчислення середнього значення
double mean(List<double> list) {
  var sum = 0.0;
  for (var i in list) {
    sum += i;
  }
  return (sum / list.length);
}
double meanFirstHalf(List<double> list, int endIndex) {
  var sum = 0.0;
  for (var i = 0; i < endIndex; i++){
    sum += list[i];
  }
  return (sum/endIndex);
}
double meanSecondHalf(List<double> list, int beginIndex) {
  var sum = 0.0;
  for (var i = beginIndex; i < list.length; i++){
    sum += list[i];
  }
  if (beginIndex == list.length){
    return sum;
  }
  if (list.length == 1) {
    return 0;
  }
  return (sum/(list.length — beginIndex));
}
```

```
// Обчислення стандартного відхилення
double standartDeviation(List<double> list, meanAccX) {
  var deviation = 0.0;
  for (var i in list) {
    deviation += pow((i — meanAccX), 2);
  }
  deviation = sqrt(deviation / (list.length — 1));
  return deviation;
}

// Обчислення діапазону значень
double range(List<double> list) {
  return list.reduce(max) — list.reduce(min);
}

// Обчислення 16 характеристик
calculateUnknown(List<Map<String, dynamic>> list) {
  List<double> listOfValues = [];
  var _meanAccX,
      _meanAccY,
      _meanOriX,
      _meanOriY,
      _standartDeviationAccX,
      _standartDeviationAccY,
      _standartDeviationOriX,
      _standartDeviationOriY,
      _rangeAccX,
      _rangeAccY,
      _meanFirstHalfAccX,
      _meanSecondHalfAccX,
      _maxOriX,
      _maxOriY,
      _minAccY,
      _duration;
```

```
// Acc.X
  for (var i in list) {
    listOfValues.add(i["acc.x"]);
  }
  _meanAccX = mean(listOfValues);
  _standartDeviationAccX = standartDeviation(listOfValues, _meanAccX);
  _rangeAccX = range(listOfValues);
  _meanFirstHalfAccX = meanFirstHalf(listOfValues, (listOfValues.length / 2).floor());
  _meanSecondHalfAccX = meanSecondHalf(
      listOfValues, (listOfValues.length / 2).floor() + 1);
// Acc.Y
  listOfValues.clear();
  for (var i in list) {
    listOfValues.add(i["acc.y"]);
  }
  _meanAccY = mean(listOfValues);
  _standartDeviationAccY = standartDeviation(listOfValues, _meanAccY);
  _rangeAccY = range(listOfValues);
  _minAccY = listOfValues.reduce(min);
// Ori.X
  listOfValues.clear();
  for (var i in list) {
    listOfValues.add(i["ori.x"]);
  }
  _meanOriX = mean(listOfValues);
  _standartDeviationOriX = standartDeviation(listOfValues, _meanOriX);
  _maxOriX = listOfValues.reduce(max);
// Ori.Y
  listOfValues.clear();
  for (var i in list) {
    listOfValues.add(i["ori.y"]);
  }
  _meanOriY = mean(listOfValues);
  _standartDeviationOriY = standartDeviation(listOfValues, _meanOriY);
  _maxOriY = listOfValues.reduce(max);
```

```
  _duration = list.last["time"] — list.first["time"];
  var _data = {
    "Range_Acc_X": _rangeAccX,
    "Range_Acc_Y": _rangeAccY,
    "Mean_Acc_X": _meanAccX,
    "Mean_Acc_Y": _meanAccY,
    "Mean_Ori_X": _meanOriX,
    "Mean_Ori_Y": _meanOriY,
    "St_Deviation_Acc_X": _standartDeviationAccX,
    "St_Deviation_Acc_Y": _standartDeviationAccY,
    "St_Deviation_Ori_X": _standartDeviationOriX,
    "St_Deviation_Ori_Y": _standartDeviationOriY,
    "Mean_first_half_Acc_X": _meanFirstHalfAccX,
    "Mean_second_half_Acc_X": _meanSecondHalfAccX,
    "Max_Ori_X": _maxOriX,
    "Max_Ori_Y": _maxOriY,
    "Min_Acc_Y": _minAccY,
    "Duration": _duration,
  };
  return _data;
}

// Метод k-найближчих сусідів
String KNN(_dataOfUnknown, _dataModels) {
  int k = 4;
  List<double> dist = [];
  bool equalClassifications = false;
  String classification;
  for (var i in _dataModels) {
    dist.add(calculateDistance(_dataOfUnknown, i));
  }

  var indexes = [];
  var distances = [];
  for (var j = 0; j < k; j++) {
```

```
    var min = 1000.0;
    for (var i in dist) {
      if (i < min && i >= 0)
        min = i;
    }
    indexes.add(dist.indexOf(min));
    distances.add(min);
    dist[dist.indexOf(min)] = -1;
  }
  for (var i = 1; i < _dataModels.length; i++) {
    if (_dataModels[i]['Classification'] ==
        _dataModels[i — 1]['Classification']) {
      equalClassifications = true;
    } else {
      equalClassifications = false;
      break;
    }
  }
  if (equalClassifications) {
    classification = _dataModels[0]['Classification'];
  } else {
    classification = vote(indexes, distances, _dataModels);
  }
  return classification;
}

// Розрахунок відстані до певного типу маневру
double calculateDistance(dataOfUnknown, model) {
  var dist = 0.0;
  for (var key in model.keys) {
    if (key != "Classification") {
      dist += pow((dataOfUnknown[key] — model[key]), 2);
    }}
  dist = sqrt(dist);
  return dist; }
```

```
// Розрахунок голосів
String vote(List indexes, List distances, _dataModels) {
  List<Map<String, dynamic>> votes = [];
  String classification;

  for (var i in distances) {
    votes.add({
      "vote": (1 / pow(i, 2)),
      "classification": _dataModels[indexes[distances.indexOf(i)]]
          ['Classification']
    });
  }
  for (var i = 0; i < votes.length — 1; i++) {
    for (var j = 1; j < votes.length; j++) {
      if (i != j) {
        if (votes[i]['classification'] == votes[j]['classification']) {
          votes[i]['vote'] += votes[j]['vote'];
          votes.removeAt(j);
        }
      }
    }
  }

  var biggestVote = votes[0]['vote'];
  classification = votes[0]['classification'];
  for (var i = 1; i < votes.length; i++) {
    if (votes[i]['vote'] > biggestVote) {
      biggestVote = votes[i]['vote'];
      classification = votes[i]['classification']; }}
  return classification;
}
```

sensorsPage.dart

```
// Розрахунок середнього значення двох змінних
double mean(a, b) {
    return (a + b) / 2;  }

// Розрахунок стандартного відхилення двох змінних
 double stDeviation(a, b, mean) {
    return sqrt(pow((a — mean), 2) + pow((b — mean), 2));
  }

// Функція аналізу поїздки
 Future<void> _analysData() async {
    List<dynamic> _abnormalClassifications = [];
    List<Map<String, dynamic>> _smoothedData = <Map<String, dynamic>>[];
    _smoothedData = smoothingSMAUnknown(_data);
    bool startAbnormal = false;
    int abnormalBeginIndex = 0;
    int abnormal = 0;

    var _dataModels =
        json.decode(await rootBundle.loadString('models/behaviorModels.json'));
    for (var i = 1; i < _smoothedData.length; i++) {
      if (stDeviation(_smoothedData[i]["acc.x"], _smoothedData[i — 1]["acc.x"],
              mean(_smoothedData[i]["acc.x"], _smoothedData[i — 1]["acc.x"])) >
          0.3) {
        if (startAbnormal != true && i != _smoothedData.length — 1) {
          startAbnormal = true;
          abnormalBeginIndex = i — 1;
        }
        if (i == _smoothedData.length — 1 && startAbnormal == true) {
          startAbnormal = false;
          abnormal += 1;
          _abnormalClassifications.add(classifyAbnormal(
              _smoothedData, _dataModels, abnormalBeginIndex, i)); }
      } else if (startAbnormal == true && i == _smoothedData.length — 1) {
```

```
    startAbnormal = false;
    abnormal += 1;
    _abnormalClassifications.add(classifyAbnormal(
        _smoothedData, _dataModels, abnormalBeginIndex, i));
  } else if (startAbnormal == true) {
    if (stDeviation(
          _smoothedData[i]["acc.x"],
          _smoothedData[i + 1]["acc.x"],
          mean(
              _smoothedData[i]["acc.x"], _smoothedData[i + 1]["acc.x"])) >
        0.23) {
    } else if (i != _smoothedData.length — 2 &&
        stDeviation(
            _smoothedData[i + 1]["acc.x"],
            _smoothedData[i + 2]["acc.x"],
            mean(_smoothedData[i + 1]["acc.x"],
                _smoothedData[i + 2]["acc.x"])) >
          0.23) {
    } else {
      startAbnormal = false;
      abnormal += 1;
      _abnormalClassifications.add(classifyAbnormal(
          _smoothedData, _dataModels, abnormalBeginIndex, i));
    }}
  if (_smoothedData[i]['time'] % 60 == 0 || i == _smoothedData.length — 1) {
    allTripAbnormal.add(abnormal);
    abnormal = 0;
  }
}
if (_abnormalClassifications.length > 0) {
  for (var ab in _abnormalClassifications) {
    ab == "Weaving"
        ? _weaving++
        : ab == "Swerving"
            ? _swerving++
```

```
            : ab == "Fast U-turn" ? _fastUTurn++ : _suddenBraking++; }}}

// Розрахунок голосів
Future<void> showProfile(BuildContext context, args) {
   return showDialog<void>(
     context: context,
     barrierDismissible: false, // user must tap button!
     builder: (BuildContext context) {
       final themeNotifier = Provider.of<ThemeNotifier>(context);
       var _darkTheme = (themeNotifier.getTheme() == darkTheme);
       var _height = MediaQuery.of(context).size.height;
       return AlertDialog(
         title: Text(
           'Behavior Profiling',
           style: TextStyle(
               color: _darkTheme ? Colors.white : Color(0xFF2a4848),
               fontFamily: "Montserrat",
               fontWeight: FontWeight.w300,
               fontSize: _height * 0.022),
         ),
         content: SingleChildScrollView(
           child: ListBody(
             children: <Widget>[getProfile(_height, _darkTheme)],
           ),
         ),
         actions: <Widget>[
           FlatButton(
             child: Text(
               'OK',
               style: TextStyle(
                   color: _darkTheme ? Colors.white : Color(0xFF2a4848),
                   fontFamily: "Montserrat",
                   fontWeight: FontWeight.w300,
                   fontSize: _height * 0.022),
             ),
```

```
            onPressed: () {
              Navigator.of(context).pop();
              Navigator.pushNamed(context, TripAnalysPage.routeName,
                  arguments: PassToTripAnalysArgs(args.auth, args.userId,
                      _weaving, _swerving, _fastUTurn, _suddenBraking));
            },
          ),
        ],
      );
    },
  );
 }

// Віджет поведінки
  Widget getProfile(_height, _darkTheme) {
    double dsi, dsiAverage = 0;
    for (var i in allTripAbnormal) {
      dsi = 1 — 0.25 * i;
      dsiAverage += dsi;
    }
    dsiAverage /= allTripAbnormal.length;
    return Column(
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          RichText(
            text: TextSpan(
                style: TextStyle(
                    color: _darkTheme ? Colors.white : Color(0xFF336666),
                    fontFamily: "Montserrat",
                    fontWeight: FontWeight.w300,
                    fontSize: _height * 0.022),
                children: <TextSpan>[
                  TextSpan(
                      text: allTripAbnormal.length > 0
```

```
        ? "You are "
        : "No data to analys behavior"),
  TextSpan(
    text: allTripAbnormal.length <= 0
        ? null
        : dsiAverage >= 0.9
            ? " a VERY SAFE "
            : dsiAverage >= 0.7
                ? "a SAFE "
                : dsiAverage >= 0.5
                    ? "an AGGRESSIVE "
                    : "a VERY AGGRESSIVE ",
    style: TextStyle(
        color: dsiAverage >= 0.9
            ? Colors.green[700]
            : dsiAverage >= 0.7
                ? Colors.green[300]
                : dsiAverage >= 0.5
                    ? Colors.orange
                    : Colors.red,
        fontFamily: "Montserrat",
        fontWeight: FontWeight.w900,
        fontSize: _height * 0.022),
  ),
  TextSpan(
      text: allTripAbnormal.length <= 0
          ? null
          : dsiAverage >= 0.9
              ? "driver. Keep it that way!"
              : dsiAverage >= 0.7
                  ? "driver, but not ideal. Try not to do dangerous maneuvers."
                  : dsiAverage >= 0.5
                      ? "driver. Calm down and drive car carefully!"
                      : "driver. Calm down! Someone is waiting for you at home!")
]),
```

```
        ),
      ]);
}

// SMA — Simple Moving Average
List<Map<String, dynamic>> smoothingSMAUnknown(data) {
  List<Map<String, dynamic>> smoothed = <Map<String, dynamic>>[];
  double smoothedAccX, smoothedAccY, smoothedOriX, smoothedOriY;
  print("Smooting");
  for (var i = 1; i < data.length; i++) {
    smoothedAccX = mean(data[i — 1]["acc.x"], data[i]["acc.x"]);
    smoothedAccY = mean(data[i — 1]["acc.y"], data[i]["acc.y"]);
    smoothedOriX = mean(data[i — 1]["ori.x"], data[i]["ori.x"]);
    smoothedOriY = mean(data[i — 1]["ori.y"], data[i]["ori.y"]);
    smoothed.add({
      "acc.x": smoothedAccX,
      "acc.y": smoothedAccY,
      "ori.x": smoothedOriX,
      "ori.y": smoothedOriY,
      "time": data[i]["time"]
    });
  }
  return smoothed;
}
```

ДОДАТОК В

Інструментальні засоби розпізнавання поведінки людини з використанням сенсорів мобільного телефону

Опис програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ6291_20Б 13-1

Аркушів 10

Київ — 2020

АНОТАЦІЯ

Додаток містить опис мобільного застосунку для виявлення та ідентифікації нетипової поведінки при керуванні транспортним засобом, що виконує деякі завдання, визначенні в розділі 1, а саме:

— Збереження даних поїздок.
— Фільтрація показників сенсорів здійсненної поїздки для подальшого аналізу.
— Знаходження початку і кінця небезпечних маневрів.
— Розрахунок 16 характеристик на кожній ділянці, яка містить певний небезпечний маневр.
— Класифікація небезпечного маневру.
— Розпізнавання поведінки водія.

Мобільний застосунок розроблений мовою програмування Dart з використанням технології Flutter у редакторі вихідного коду Visual Studio Code.

ЗМІСТ

ЗАГАЛЬНІ ВІДОМОСТІ

У додатку міститься опис основних компонентів мобільного застосунку для виявлення та ідентифікації нетипової поведінки при керуванні транспортним засобом, що виконує деякі завдання, визначенні в розділі 1. Додаток Б містить програмний код цих компонентів.

Для роботи мобільного застосунку потрібно мати мобільний пристрій на платформі Android версії Jelly Bean (4.1.x) і вище, або iOS версії 8 і вище. Також необхідно мати підключення до Інтернету.

Мобільний застосунок розроблений мовою програмування Dart з використанням технології Flutter у редакторі вихідного коду Visual Studio Code.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблений мобільний застосунок надає користувачу наступний функціонал:

— Реєстрація/авторизація в системі.

— Запис та аналіз поїздки.

— Збереження даних поїздки.

— Аналіз поведінки при керуванні транспортним засобом.

— Перегляд загальної статистики всіх поїздок і розгорнутої статистики окремої поїздки.

— Редагування особистих даних.

— Зміна налаштувань застосунку.

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Згідно з архітектурою, застосунок складається з двох частин: моделювання поведінки водія та аналіз поведінки.

Моделювання поведінки містить наступні логічні складові: модель та сервіси.

Моделлю є файл, що містить еталонні зразки кожного небезпечного маневру. Ці зразки зберігаються як об’єкти, що містять 16 характеристик.

Сервіс — клас, що містить функцію для розрахунку цих характеристик (середнього значення, діапазону значень, максимального та мінімального значення).

Аналіз поведінки складається з наступних частин: представлення, сервіси і модель.

Модель — клас-сутність даних поїздки (покази акселерометра і гіроскопа), що зберігаються кожні півхвилини.

Представлення — класи, що відображають інформацію і структурні елементи сторінок мобільного застосунку (заголовки, іконки, кнопки і т. д.).

Сервіси — клас зв’язку з базою даних; клас розрахунків, що містить функцію реалізації SMA методу для фільтрації даних, функцію обчислення 16 характеристик, функції для реалізації KNN алгоритму, функцію класифікації поведінки.

ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Для роботи мобільного застосунку потрібно мати мобільний пристрій на платформі Android версії Jelly Bean (4.1.x) і вище, або на платформі iOS версії 8 і вище.

Смартфон має бути оснащений гіроскопом і акселерометром. Також необхідно мати підключення до Інтернету під час роботи в мобільному застосунку.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Для встановлення мобільного застосунку на iOS пристрій необхідно завантажити файл app_release.ipa, потім запустити його і натиснути кнопку “Встановити” (“Install”). Для інсталяції застосунку на пристрій на платформі Android треба завантажити файл app_release.apk, запустити його і почати встановлення, натиснувши на кнопку “Встановити” (“Install”).

ВХІДНІ ДАНІ

Вхідна інформація для моделювання поведінки водія:

— Тренувальна вибірка із зразками небезпечних маневрів (записані кожні півхвилини покази сенсорів).

Вхідні дані для аналізу поведінки водія:

— Дані для створення облікового запису (ім’я, адреса електронної пошти, пароль).

— Дані поїздки, яку потрібно проаналізувати.

— Відфільтровані дані поїздки, що передаються у функцію виокремлення характеристик.

— Виокремлені характеристики для подальшої класифікації маневру.

— Кількість небезпечних маневрів для розпізнавання поведінки.

ВИХІДНІ ДАНІ

Вихідна інформація офлайн частини (Моделювання поведінки водія):

— Модель класифікатора.

Вихідні дані онлайн частини (Аналіз поведінки водія):

— Відфільтровані дані поїздки (після функції фільтрації методом SMA).

— Виокремлені характеристики.

— Визначений клас небезпечного маневру.

— Кількість небезпечних маневрів.

— Визначена поведінка при керуванні транспортним засобом.

— Загальна статистика всіх поїздок.

— Розгорнута статистика окремої поїздки.